Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

Does Graph Attention Improve Weighted-Polling Classification on the Pima Indians Diabetes Dataset? A Nested Cross-Validation Comparison with Standard Models

Farid Morsidi

farid.mors90@gmail.com

Universiti Pendidikan Sultan Idris

Research Article

Keywords:

Posted Date: November 9th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-11160715/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

Does Graph Attention Improve Weighted-Polling Classification on the Pima Indians Diabetes Dataset? A Nested Cross-Validation Comparison with Standard Models

Farid Morsidi[1*]

[1*]Independent Researcher, Sarawak, Malaysia.

Corresponding author(s). E-mail(s): farid.mors90@gmail.com;

Abstract

Background: It is not known whether a graph attention encoder improves weighted-polling classification of small clinical tabular datasets. A GATv2 weighted-polling classifier developed for agricultural morphology data was therefore tested on the Pima Indians Diabetes Dataset (768 instances, 8 features). Its first evaluation scored nodes whose labels had trained the encoder.

Methods: Nested stratified cross-validation (10 outer folds, 5 inner folds, three repeats) was used, and every hyperparameter was chosen in the inner loop only: graph neighbourhood size, edge weighting, encoder width, attention heads, dropout, learning rate, weight decay, training epochs and the polling configuration. On identical outer folds, the tuned GATv2 classifier was compared with raw-feature weighted polling and five standard classifiers (logistic regression, RBF support vector machine, random forest, gradient boosting, k-nearest neighbours), each tuned in the same inner loop. Accuracy, macro-F1, sensitivity, specificity and area under the ROC curve (AUROC) were reported, with 95% bootstrap confidence intervals for paired differences.

Results: The tuned GATv2 classifier reached 75.22% accuracy, 71.90% macro-F1, 58.58% sensitivity, 84.13% specificity and an AUROC of 0.808. Its accuracy differed from that of raw-feature weighted polling by $\mathbf{-0.13}$ points (95% CI $\mathbf{-2.13}$ to $\mathbf{+1.87}$) and from that of the most accurate standard model, logistic

regression (76.74%), by $-\mathbf{1.52}$ points (95% CI $-\mathbf{3.47}$ to $+\mathbf{0.39}$). Its AUROC was 1.4 to 2.5 points lower than that of the other classifiers, with intervals excluding zero for four of the five standard models. The original protocol, which scored nodes used to train the encoder, had given 79.01% accuracy.

Conclusions: On this dataset, tuning the encoder under nested cross-validation did not make GATv2 weighted polling more accurate than raw-feature weighted polling or than standard classifiers. Relative to raw-feature polling, accuracy differences larger than about 2 points are excluded in either direction, but a benefit of up to about 1.9 points cannot be excluded. These findings are limited to this single dataset and do not show that graph attention cannot help on other clinical data.

Keywords: Graph Neural Networks, Weighted Polling, Nested Cross-Validation, Diabetes Classification, Small Clinical Datasets

1 Background

Diabetes mellitus is a chronic disease whose prevalence is rising worldwide, which makes early and accurate diagnosis a priority. Classifying patients from structured data with machine learning has been studied extensively, and the Pima Indians Diabetes Dataset is one of the standard benchmarks. Graph-based methods have recently attracted attention for tabular classification. One approach builds a k-nearest-neighbour similarity graph from the instances' features and trains a graph neural network to learn node representations that preserve this similarity. The Graph Attention Network Veličković et al. (2018) and its dynamic-attention variant GATv2 Brody, Alon, and Yahav (2022) suit this task because their attention mechanism can model instance-specific relationships between neighbours, learning which edges are most informative for a given query node.

The pipeline studied here, a GATv2 encoder followed by a weighted-polling KNN/naive Bayes classifier, was developed by the author for multiclass variety identification on the Dry Bean dataset, which has 13,611 instances, 7 moderately imbalanced classes and 8 clean morphological features Koklu and Ozkan (2020). That earlier work Morsidi (2026) built on the weighted-polling KNN/Bayes classifier introduced by the

author Morsidi (2024). The Dry Bean setting has clean data, a large instance count and broad class separation, and it is not known whether the same architecture benefits domains with different characteristics.

This paper applies the same architecture to a binary clinical diagnosis task, using the Pima Indians Diabetes Dataset. The dataset differs from the Dry Bean data in several key ways. It is smaller by more than an order of magnitude (768 instances against 13,611), it has a binary rather than seven-class label with a class imbalance of 65%/35%, and it contains well-documented data-quality problems: in 5 of its 8 features, missing values were recorded as zeros. These differences matter especially for graph-based methods, because they may make generalization harder.

Evaluation design matters for such a comparison. In a transductive graph pipeline, scoring nodes whose labels trained the encoder overstates performance Kapoor and Narayanan (2023), and on a small dataset a search over many configurations adds further optimism. The first version of this pipeline was evaluated in that way (Section 4.4), so the comparison is repeated here with nested cross-validation, in which every hyperparameter, including those of the encoder, is chosen without access to the folds on which performance is reported.

The aim of this study is to test, under such a protocol, whether GATv2 augmentation improves weighted-polling classification on the Pima Indians Diabetes Dataset, and to compare it with raw-feature weighted polling and standard classifiers on identical folds, using clinically relevant metrics and confidence intervals for paired differences. The contributions are:

1. **A nested cross-validation comparison** in which the encoder, the graph and the polling configuration are all tuned in the inner loop;
2. **A paired comparison on identical folds** with raw-feature weighted polling and five standard classifiers, reporting accuracy, macro-F1, sensitivity, specificity and AUROC, with 95% confidence intervals for the differences;

3. **A short quantification** of how much the original protocol overstated performance (Section 4.4);
4. **Released code** implementing the protocols (see Declarations).

The rest of this paper is organized as follows: Section 2 reviews related work, Section 3 describes the architecture, the nested protocol, the comparison models and the statistical analysis, Section 4 reports the results, Section 5 discusses them and their limitations, and Section 6 concludes.

2 Related Work

2.1 Graph Neural Networks for Tabular Data

Applying GNNs to tabular data is not trivial, because it requires choosing a graph structure to impose on data that does not inherently have one. A recent survey gives a taxonomy of this graph-construction problem and identifies the k-nearest-neighbour graph as one of the common rule-based constructions, alongside learning-based approaches such as metric-learned or fully learned adjacency matrices Li, Tsai, Chen, and Liao (2025). RF-GNN builds instance graphs from random-forest-induced proximity measures instead of pairwise distances and reports consistent improvements over classical approaches on 36 benchmark datasets Chen, Farokhi, Bladen, Karimi, and Moon (2026). Defilippo, Veltri, Liò, and Guzzi (2024) perform triage classification on patient data with graph convolutional network (GCN), GAT and GraphSAGE architectures and benchmark them against SVM and KNN classifiers trained on the same tabular data.

2.2 Domain Generalization in Graph Neural Networks

A separate line of work investigates whether GNN architectures and representations transfer across distinct graph domains instead of being retrained for each target graph.

Lachi, Azabou, Arora, and Dyer (2024) train GraphFM, a large-scale multi-graph model with a shared latent representation, to generalize across structurally different graphs, on the premise that GNNs are typically optimized for the particular graph on which they are trained. This suggests that domain generalization should not be taken for granted for GNNs and should be evaluated explicitly. The present work builds on this reasoning but targets a much smaller architecture (a two-stage GATv2 and weighted-polling classifier) and a different domain shift (from the morphology of agricultural plants to tabular clinical data), and it does not use the pretraining of that work.

2.3 Machine Learning Approaches to Pima Diabetes Classification

The Pima Indians Diabetes Dataset has an extensive classification literature. The reported classification accuracies vary widely depending on the method used, but also the preprocessing and validation strategy. Table 1 below shows a selection of results on the standard 768-instance version of the dataset. A region-based support vector machine, which uses an RBF-kernel SVM for coherent data regions and a second, polynomial-kernel SVM for the hard-to-cluster remainder, reached an average classification success rate of 82.2% Karatsiolis and Schizas (2012). A stacking ensemble of machine-learning classifiers reached 75.03% under a train-test split and 77.10% under cross-validation, while a random forest reached 79.33% and 76.81% under the same two protocols Reza, Amin, Yasmin, Kulsum, and Ruhi (2024).

Reported accuracies on this dataset depend on preprocessing and validation strategy. The studies in Table 1 span about 75–82% under different protocols, so they are not directly comparable with the nested cross-validation estimates reported here.

Table 1 Representative reported accuracies on the standard 768-instance Pima Indians Diabetes Dataset, with the validation protocol each study reports.

Method	Reported Accuracy	Validation	Source
Region-based SVM	82.2%	Average success rate; protocol not stated in abstract	Karatsiolis and Schizas (2012)
Random forest	79.33%	Train-test split	Reza et al. (2024)
Random forest	76.81%	Cross-validation (folds not stated)	Reza et al. (2024)
Stacking ensemble of ML classifiers	75.03%	Train-test split	Reza et al. (2024)
Stacking ensemble of ML classifiers	77.10%	Cross-validation (folds not stated)	Reza et al. (2024)
This study: GATv2 + weighted polling	75.22% (±0.33%)	Nested repeated 10-fold CV (5-fold inner), three repeats	—

2.4 Evaluation Leakage and Validation Practice

A survey of machine-learning-based science found leakage in papers across 17 fields Kapoor and Narayanan (2023). In a transductive graph pipeline, the features of held-out nodes may legitimately enter the graph, but using their labels in the encoder's loss is leakage. Selecting hyperparameters on the same folds that are then reported adds optimism of a different kind, which nested cross-validation is designed to remove.

2.5 Positioning of This Study

Compared with the work on graph neural networks for tabular data (Section 2.1), this study makes a direct within-architecture comparison of GATv2-augmented and raw-feature weighted polling, and also compares both with standard classifiers. Compared with the work on domain generalization (Section 2.2), it asks a narrower question: whether an existing two-stage architecture helps on one new clinical dataset when its encoder is tuned under nested cross-validation.

3 Methods

3.1 Overview

The proposed classification methodology has two steps. First, a Graph Attention Network (GATv2) encodes a relational embedding of each instance by passing messages over a k-nearest-neighbour similarity graph built from the instances' feature vectors. This yields a fixed-dimensional latent representation that captures both the instance's own characteristics and its relationship to similar instances. Second, a weighted-polling classifier assigns each instance a class label by aggregating the votes of its nearest neighbours in the latent space, using one of four voting schemes. The rest of this section describes each step and the parameters that define the binary clinical diagnosis task.

3.2 Graph Attention Networks

Graph Attention Networks (GAT) Veličković et al. (2018) generalize convolutional-type graph neural networks by learning to attend to specific nodes when propagating information through the graph. In other words, while GCN assumes that all neighbors contribute equally to the representation of a given node and relies on structural information encoded in the adjacency matrix, GAT uses a self-attention mechanism to compute a particular weight for each edge in the graph. Specifically for each edge, an attention coefficient is computed indicating how much information should be passed from the source node to the target node during the updates.

While the original GAT architecture Veličković et al. (2018) has proven successful, Brody et al. (2022) note that its attention is only statically expressive: the attention function applies a linear transformation to the inputs followed by a fixed nonlinearity, so the ranking of neighbours is the same for every query node. GATv2 removes this limitation by applying the nonlinearity before the attention vector, which makes the attention function expressive enough to rank neighbours differently for different query

nodes (dynamic attention). This capability matters for graphs built from feature similarities, as in this study, because the importance of a neighbour can depend on the characteristics of the query node.

3.3 Mathematical Formulation

For a node i with feature vector h_i and neighbourhood $N(i)$ defined by the constructed k-NN graph, the standard GAT attention coefficient between i and a neighbour j is computed as:

$$e_{ij} = \text{LeakyReLU}\left(a^\top \left[Wh_i \parallel Wh_j\right]\right) \tag{1}$$

where W is a learned linear transformation shared across all nodes, a is a learned attention weight vector and $\parallel$ denotes concatenation. GATv2 modifies this by applying the non-linearity before the attention vector a, so that the linear map W acts on the concatenated pair:

$$e_{ij} = a^\top \text{LeakyReLU}\left(W \left[h_i \parallel h_j\right]\right) \tag{2}$$

In both formulations, attention coefficients are normalized across each node's neighborhood using a softmax:

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k \in N(i)} \exp(e_{ik})} \tag{3}$$

Where the constructed graph contains edge weights (Section 3.7 – inverse-distance weighting is one of the two edge schemes we evaluate), these are input to the attention layer as edge features and added to the attention score as per the implementation of GATv2 layers (PyTorch Geometric's GATv2Conv, `edge_dim`=1); this is an implementation detail of how the edge weights contribute to the attention coefficients, rather than a change to the formulation described above.

The updated node representation takes the form of a weighted sum of neighbour features, passed through a non-linearity σ (ELU in this work):

$$h_i' = \sigma\left(\sum_{j \in N(i)} \alpha_{ij} W h_j\right) \tag{4}$$

and multi-head attention concatenates K independently learned attention heads to stabilize training and capture different relational patterns simultaneously:

$$h_i' = \Big\Vert_{k=1}^{K} \sigma\left(\sum_{j \in N(i)} \alpha_{ij}^k W^k h_j\right) \tag{5}$$

The baseline three-layer architecture (Section 3.7) stacks two multi-head layers with $K = 4$ heads and concatenation, followed by a final single-head layer without concatenation that projects to the 16-dimensional embedding; the hidden width and the number of heads are tuned in the nested search.

3.4 Weighted-Polling Classification

Given a query instance's learned embedding, its k nearest neighbours (by Euclidean distance in embedding space) among the labelled instances are ranked by distance, with rank $r = 0$ denoting the nearest neighbour. Four voting schemes are defined to convert this ranked neighbour set into a class prediction. Let y_r denote the class label of the neighbour at rank r, and $\mathbf{1}[\cdot]$ the indicator function.

Method 1 (*Normal*): each of the k-neighbours casts one equal vote:

$$\text{score}(c) = \sum_{r=0}^{k-1} \mathbf{1}[y_r = c] \tag{6}$$

Method 2 (*Weighted*): closer neighbours are weighted more heavily by their inverse rank:

$$\text{score}(c) = \sum_{r=0}^{k-1} (k - r)\, \mathbf{1}[y_r = c] \tag{7}$$

Method 3 (*Weighted-Squared*): the rank-based weight is squared to further emphasize the closest neighbours:

$$\text{score}(c) = \sum_{r=0}^{k-1} (k-r)^2 \, \mathbf{1}[y_r = c] \tag{8}$$

Method 4 (*Naive-Bayes frequency-weighted):* raw votes are normalized by each class's prior frequency N_c in the training set, correcting for class imbalance:

$$\text{score}(c) = \frac{\sum_{r=0}^{k-1} \mathbf{1}[y_r = c]}{N_c} \tag{9}$$

The predicted class is $\hat{y} = \arg\max_c \text{score}(c)$. If two or more classes are tied for the highest score, the neighbourhood is expanded by one additional (farther) neighbour and re-scored; this expansion repeats until a unique winner emerges or all labelled instances have been included, in which case the label of the single nearest neighbour ($r = 0$) is returned as a final tie-break (Figure 1).

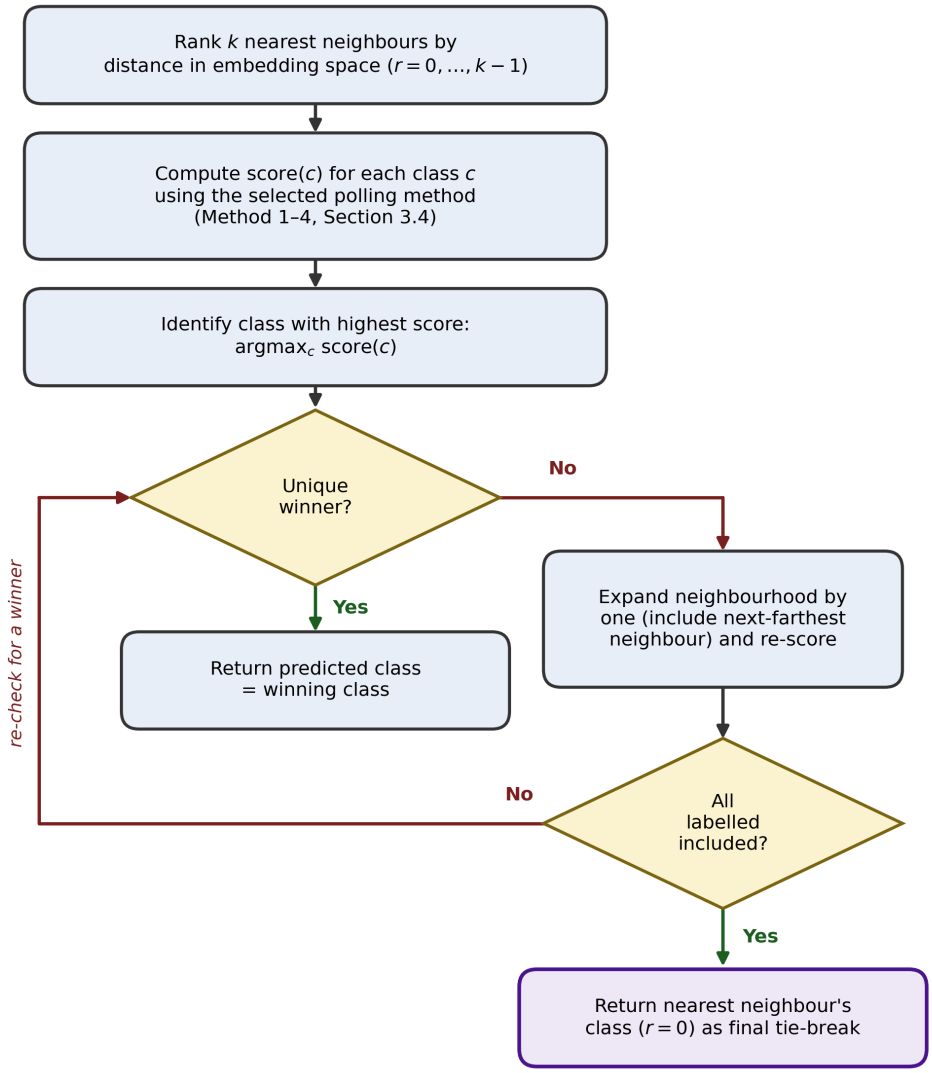


Fig. 1 Weighted-polling tie-break decision logic

3.5 Proposed Pipeline

Figure 2 summarizes the entire processing chain from raw input to final prediction, as applied to the binary clinical diagnosis task considered in this paper. Each raw

8-dimensional instance is first pre-processed by imputing missing values and normalizing the data with a z-score (Section 3.6). The normalized instances are then used to construct a k -nearest-neighbour similarity graph (Section 3.7). This graph is subsequently encoded with the three-layer GATv2 architecture (Section 3.3), yielding a 16-dimensional relational embedding per instance. Finally, the votes of each instance's neighbours within this embedding are aggregated via the weighted-polling classifier (Section 3.4) to predict its class, with the voting scheme and number of neighbours selected in the inner loop of the nested search (Sections 3.7 and 3.8).

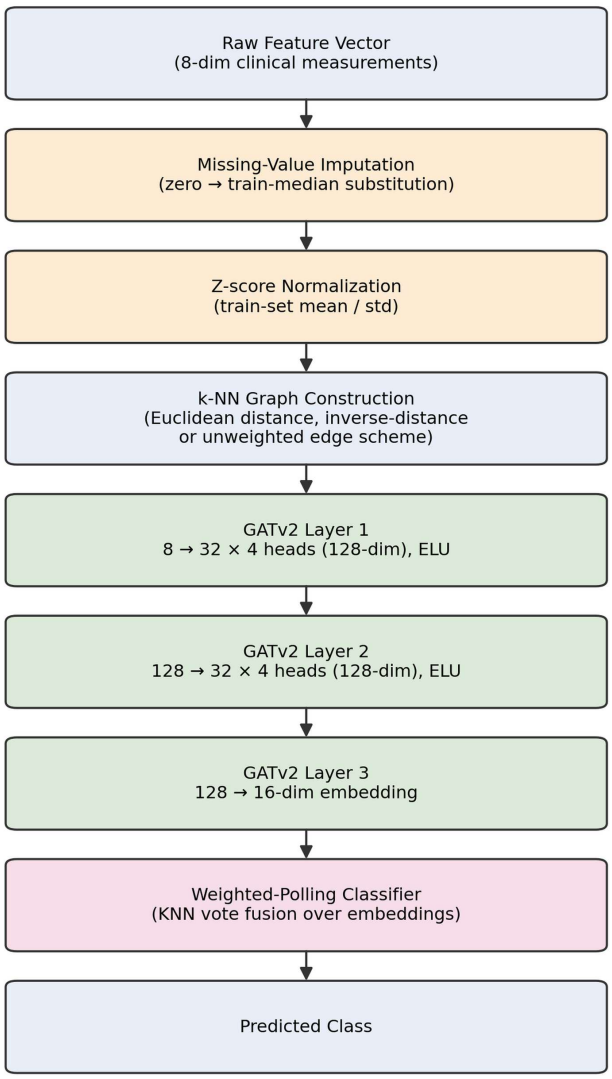


Fig. 2 End-to-end pipeline: raw features are cleaned and normalized, embedded into a 16-dimensional relational representation via a 3-layer GATv2, and classified via weighted-polling over that embedding space

3.6 Dataset

For this analysis, the Pima Indians Diabetes Dataset (768 instances, 8 numeric features, binary Outcome) was used. The dataset originates from the National Institute of

Diabetes and Digestive and Kidney Diseases Smith, Everhart, Dickson, Knowler, and Johannes (1988) and describes female patients of Pima Indian heritage aged at least 21 years; the copy analysed here was obtained from a widely used GitHub repository Brownlee (2019). Outcome classes are imbalanced with 500 (65.1%) negative and 268 (34.9%) positive instances. Additionally, 5 out of 8 features are known to have records with zero values that were used to represent missing data in the original study, which is in contradiction with the actual physiological ranges of these variables. This can be seen in Table 2 below.

Table 2 Zero-encoded missing values by feature.

Feature	**Zero-value count (of 768)**
Glucose	5
Blood Pressure	35
Skin Thickness	227
Insulin	374
BMI	11

To prevent leakage into imputation statistics from the test set, zeroes in *Glucose*, *Blood Pressure*, *Skin Thickness*, *Insulin*, and *BMI* were considered as missing values and replaced with the median value of that feature taken from the training rows of the respective fold only (Section 3.8). In total, 652 of 6144 feature values (10.6%) in the dataset have been imputed in this manner. The imputation of missing values is very prevalent across *Insulin* (48.7%) and *Skin Thickness* (29.6%).

3.7 Encoder and Search Space

The baseline encoder is the three-layer GATv2 described above ($8 \to 32 \times 4$ heads $\to 32 \times 4 \to 16$, ELU activations, dropout 0.3, Adam with learning rate 0.01 and weight decay 5×10^{-4}, 75 epochs, class-weighted cross-entropy), whose settings were taken from the earlier Dry Bean study Morsidi (2026). Here the encoder was tuned.

Twenty-four candidates were evaluated: the baseline encoder, and 23 candidates drawn at random from the following space: $k_{graph} \in \{3, 5, 7, 10, 15\}$, edge weighting {unweighted, weights $1/(1+d)$ for Euclidean distance d}, hidden width {16, 32, 64}, attention heads {2, 4}, dropout {0.1, 0.3, 0.5}, learning rate {0.01, 0.005}, weight decay $\{5\times10^{-4}, 5\times10^{-3}\}$ and training epochs {50, 75, 150}. The embedding dimension was fixed at 16. For each candidate, the 20 polling configurations ($k_{poll} \in \{3, 5, 7, 10, 15\}$ and polling method {1,2,3,4}) were scored on the same embeddings. All experiments were run in Python 3.11 with PyTorch 2.12.0, PyTorch Geometric 2.8.0 and scikit-learn 1.9.0.

3.8 Nested Cross-Validation

Performance was estimated with repeated nested stratified cross-validation (Figure 3). The outer loop used 10 folds, repeated with three seeds (0–2) that change the fold assignment and the GATv2 initialization. Within each outer-training set, an inner stratified 5-fold loop selected all hyperparameters: the candidate encoder and graph settings and the polling configuration for GATv2, and the polling configuration for the raw-feature baseline. The selection criterion was the pooled inner out-of-fold accuracy. The selected configuration was then refitted on the whole outer-training set and scored once on the outer-test fold. Every model was scored on the same outer folds.

In every training step, imputation medians and z-score parameters were computed from the training rows of that split only. The k-nearest-neighbour graph was built over all instances, a transductive setting in which held-out instances contribute features but never labels; this also applies in the inner loop, where the features of the outer-test instances are present in the graph. The GATv2 loss was computed on training nodes only, and polling votes, including class frequencies, were drawn from training instances only.

3.9 Comparison Models and Metrics

The tuned GATv2 classifier (GATv2-tuned) was compared with three groups of models. First, GATv2-default used the baseline encoder with only the polling configuration tuned. Second, raw-feature weighted polling used the z-scored features with the same 20 polling configurations. Third, five standard classifiers were tuned in the same inner loop: logistic regression ($C \in \{0.01, 0.1, 1, 10, 100\}$), a support vector machine with RBF kernel ($C \in \{0.1, 1, 10\}$, $\gamma \in \{\text{scale}, 0.01, 0.1\}$), a random forest (200 trees, maximum depth {3, 5, none}, minimum leaf size {1, 3, 5}), gradient boosting ({50, 100, 200} trees, learning rate {0.03, 0.1}, depth {2, 3}) and k-nearest neighbours ($k \in \{3, 5, 7, 10, 15, 21, 31\}$, uniform or distance weights).

Accuracy, macro-F1, sensitivity (recall of the diabetic class), specificity and AUROC were computed on the pooled outer-test predictions of each repeat. AUROC used predicted probabilities for logistic regression, random forest, gradient boosting and k-nearest neighbours, the decision function for the support vector machine, and the weighted share of diabetic votes among the k neighbours for the polling models. Because that share takes few distinct values, the AUROC of the polling models is coarse.

3.10 Statistical Analysis

For each metric, the paired difference between GATv2-tuned and every other model was averaged over repeats. Its 95% confidence interval was obtained by a percentile bootstrap that resamples the 768 instances (10,000 resamples; 1,000 for AUROC). Exact McNemar tests on paired correctness were computed per repeat as a descriptive check. The intervals reflect the sampling of instances only; they do not capture the variance from redrawing the training data, and the repeats resample the same 768 instances and are not independent. Intervals were used to state which differences can be excluded, and a non-significant test was not interpreted as equivalence.

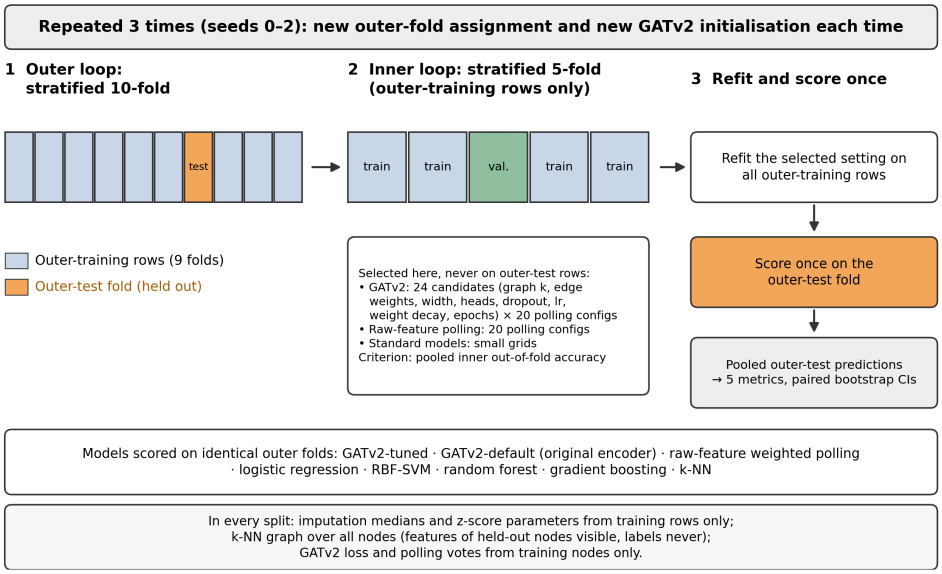


Fig. 3 Nested cross-validation protocol. Each repeat uses a new stratified 10-fold outer split. Inside each outer-training set, a stratified 5-fold inner loop selects the encoder, graph and polling settings (GATv2) or the hyperparameters of each comparison model, using only the outer-training rows. The selected setting is refitted on all outer-training rows and scored once on the outer-test fold. All models are scored on identical outer folds

4 Results

4.1 Nested Cross-Validation Performance

Table 3 gives the performance of all models on identical outer folds. GATv2 weighted polling with a tuned encoder reached 75.22% (±0.33%) accuracy and 71.90% (±0.49%) macro-F1, against 75.35% (±0.83%) and 72.02% (±1.01%) for raw-feature weighted polling. Across the eight models, accuracy lay between 75.22% and 76.74%, and logistic regression was the highest (76.74%, macro-F1 72.83%, AUROC 0.833). Sensitivity ranged from 53.23% (SVM) to 62.56% (GATv2 with the original encoder) and specificity from 82.07% to 88.07%, so every model identified fewer than two thirds of the diabetic cases. The two GATv2 variants had the lowest AUROC (0.807 and 0.808), against 0.822 to 0.833 for raw-feature polling and the standard models. The AUROC of the polling models is coarse, because the vote share takes few distinct values.

Table 3 Nested cross-validation performance, mean (SD) over three repeats of 10-fold outer cross-validation, on pooled outer-test predictions. Values are percentages, except AUROC. Sensitivity is the recall of the diabetic class. The SD reflects fold assignment and initialization only.

Model	Accuracy	Macro-F1	AUROC
GATv2 (tuned)	75.22 (0.33)	71.90 (0.49)	0.808 (0.009)
GATv2 (original encoder)	75.26 (0.81)	72.52 (0.84)	0.807 (0.009)
Raw-feature polling	75.35 (0.83)	72.02 (1.01)	0.822 (0.003)
Logistic regression	76.74 (0.40)	72.83 (0.47)	0.833 (0.003)
SVM (RBF kernel)	75.87 (0.30)	71.61 (0.41)	0.832 (0.002)
Random forest	76.09 (0.53)	72.34 (0.68)	0.827 (0.004)
Gradient boosting	75.82 (0.89)	71.94 (1.13)	0.822 (0.001)
k-nearest neighbours	75.61 (0.91)	71.93 (1.10)	0.826 (0.003)
	Sensitivity	**Specificity**	
GATv2 (tuned)	58.58 (2.08)	84.13 (1.10)	
GATv2 (original encoder)	62.56 (0.78)	82.07 (0.92)	
Raw-feature polling	58.58 (1.87)	84.33 (0.64)	
Logistic regression	55.60 (0.65)	88.07 (0.31)	
SVM (RBF kernel)	53.23 (1.08)	88.00 (0.53)	
Random forest	56.34 (1.63)	86.67 (0.76)	
Gradient boosting	55.35 (2.37)	86.80 (1.06)	
k-nearest neighbours	56.47 (1.84)	85.87 (0.76)	

4.2 Paired Differences

Table 4 gives the paired differences between the tuned GATv2 classifier and each other model. The accuracy difference from raw-feature weighted polling was −0.13 points (95% CI −2.13 to +1.87), so a difference larger than about two points in either direction is excluded, but a benefit of up to about 1.9 points cannot be. Against the standard models, the point estimates for accuracy were 0.39 to 1.52 points lower for GATv2, with upper confidence limits between +0.39 and +1.43 points; all accuracy and macro-F1 intervals included zero. The AUROC of the tuned GATv2 classifier was lower than that of logistic regression, SVM, random forest and k-nearest neighbours, with intervals excluding zero (−2.46, −2.42, −1.93 and −1.84 points of AUROC×100), but not for gradient boosting (−1.43 [−3.12, +0.25]) or raw-feature polling (−1.42 [−3.00, +0.30]). Relative to the standard models, GATv2 showed higher sensitivity and lower specificity (for example +5.35 and −3.87 points against the SVM), which indicates a different operating point and not better discrimination. Exact McNemar

tests per repeat gave $p > 0.49$ against raw-feature polling (0.85, 0.65, 1.00) and against the original encoder (0.55, 1.00, 0.50); these were not interpreted as evidence of equivalence. Because 35 unadjusted intervals are reported, isolated exclusions of zero should be read with caution.

Tuning the encoder did not improve accuracy: relative to the original encoder, the tuned classifier differed by −0.04 points in accuracy (95% CI −1.69 to +1.61) and −0.62 points in macro-F1 (−2.44 to +1.21). It lowered sensitivity (−3.98 points, 95% CI −7.06 to −0.94) and raised specificity (+2.07 points, 95% CI +0.13 to +3.98).

Table 4 Paired differences, GATv2 weighted polling with a tuned encoder minus each comparison model, in percentage points (AUROC in AUROC×100), averaged over three repeats, with 95% percentile bootstrap intervals over the 768 instances in brackets. The intervals do not include variance from re-drawing the training data, and none is adjusted for multiplicity.

Comparison model	**Accuracy**	**Macro-F1**	**AUROC**
GATv2 (original encoder)	−0.04 [−1.69, +1.61]	−0.62 [−2.44, +1.21]	+0.09 [−1.34, +1.56]
Raw-feature polling	−0.13 [−2.13, +1.87]	−0.12 [−2.41, +2.17]	−1.42 [−3.00, +0.30]
Logistic regression	−1.52 [−3.47, +0.39]	−0.93 [−3.07, +1.19]	−2.46 [−3.94, −0.90]
SVM (RBF kernel)	−0.65 [−2.39, +1.04]	+0.29 [−1.67, +2.22]	−2.42 [−3.79, −1.09]
Random forest	−0.87 [−2.82, +1.04]	−0.45 [−2.70, +1.79]	−1.93 [−3.45, −0.35]
Gradient boosting	−0.61 [−2.56, +1.35]	−0.04 [−2.24, +2.17]	−1.43 [−3.12, +0.25]
k-nearest neighbours	−0.39 [−2.30, +1.43]	−0.03 [−2.24, +2.14]	−1.84 [−3.23, −0.35]

Comparison model	**Sensitivity**	**Specificity**
GATv2 (original encoder)	−3.98 [−7.06, −0.94]	+2.07 [+0.13, +3.98]
Raw-feature polling	+0.00 [−3.91, +4.04]	−0.20 [−2.41, +2.06]
Logistic regression	+2.99 [−0.25, +6.20]	−3.93 [−6.34, −1.54]
SVM (RBF kernel)	+5.35 [+2.41, +8.37]	−3.87 [−5.97, −1.86]
Random forest	+2.24 [−1.53, +6.03]	−2.53 [−4.78, −0.35]
Gradient boosting	+3.23 [−0.39, +6.80]	−2.67 [−5.04, −0.34]
k-nearest neighbours	+2.11 [−1.72, +5.95]	−1.73 [−3.76, +0.27]

4.3 Selected Configurations

The inner loop selected a non-default encoder in 29 of the 30 outer folds, but the selections varied: k_{graph} was 7 in 14 folds, 3 in 8, 5 in 4, and 10 or 15 in 2 each; edges were unweighted in 17 folds and inverse-distance in 13; dropout was 0.5 in 15 folds and 0.3 in 13; and weight decay was 5×10^{-3} in 20. Polling used majority voting (method 1) in 24 of 30 folds, most often with $k_{poll} = 15$ (15 folds) or 10 (8 folds). No single encoder setting dominated.

4.4 Evaluation Leakage in the Original Protocol

The first version of this pipeline trained the encoder once on a fixed stratified 80% mask (614 nodes) and then scored all 768 nodes by leave-one-out polling, so most scored nodes had trained the encoder. For the configuration chosen by its own search (k_{graph}=7, unweighted edges, k_{poll}=15, method 1), this gave 79.01% (±0.52%) accuracy and 76.93% (±0.59%) macro-F1, against 76.17% and 72.78% for raw features. In the seed-0 reproduction, accuracy was 81.43% on the 614 nodes inside the training mask and 70.13% on the 154 nodes outside it (a binomial standard error of about 3.7 points for the latter, so this is a diagnostic and not an estimate of generalization). When the encoder was retrained inside every fold, the same configuration gave 75.96% (±0.61%) accuracy and 73.27% (±0.73%) macro-F1 over repeated 10-fold cross-validation, and the configuration chosen by the leakage-free search gave 76.20% accuracy, against 76.25% for raw-feature polling. The nested estimates in Section 4.1 are 75.22% for GATv2 and 75.35% for raw-feature polling. They are 0.7 to 1.0 points lower than these fixed-configuration estimates, which is consistent with no longer selecting the configuration on the evaluation folds, although the fold assignments and the number of repeats also differ, and they are about 3.8 points below the 79.01% of the original protocol. In a label-shuffle control (20 permutations), mean accuracy was 61.73% (SD 2.01) under the original protocol and 60.45% (SD 1.56) under retraining,

both below the 65.10% majority-class rate; one permutation exceeded it under the original protocol (65.76%) and none under retraining.

5 Discussion

Under nested cross-validation with a tuned encoder, GATv2 augmentation did not improve weighted-polling classification on the Pima dataset. The tuned GATv2 classifier and raw-feature weighted polling differed by -0.13 points in accuracy and -0.12 points in macro-F1, with intervals of about ± 2 points, so a difference larger than about two points in either direction is excluded, but a small benefit, up to about 1.9 points of accuracy, cannot be. The encoder also did not bring polling to the level of the standard classifiers: its AUROC was lower than that of four of the five (Table 4), and logistic regression had the highest point estimate for accuracy. Tuning the encoder did not help either. A non-default encoder was chosen in 29 of 30 outer folds, yet the tuned and the original encoder had the same outer-test accuracy, and tuning mainly moved the operating point from sensitivity towards specificity.

The clinical reading is cautious. Sensitivity was between 53% and 63% for every model, so each would miss about 37% to 47% of diabetic cases at its default decision rule. A screening use would need an operating point chosen for sensitivity, for example by adjusting a decision threshold or class weights, which this study did not do; the polling models in particular have no tunable threshold, since the vote share takes few distinct values.

Why the encoder adds nothing here is not established by this study. One hypothesis, which was not tested, is that a k-nearest-neighbour graph built from eight noisy features carries little information beyond the features themselves, so that message passing cannot improve on polling directly over the z-scored features. The dataset is

small (768 instances), 10.6% of its values are imputed, and its two classes overlap substantially. The selected encoder and graph settings varied across folds, which suggests that no setting was clearly better on this dataset.

The nested accuracies (75.2% to 76.7%) lie at the low end of the range in Table 1, so this study makes no claim to state-of-the-art performance. The higher published figures come from studies with different preprocessing and validation protocols, and they are not directly comparable with a repeated nested cross-validation estimate. For the same architecture, the original protocol reported 79.01%, about 3.8 points above the nested estimate, mainly because the encoder had been trained on the labels of most of the nodes it was scored on, and in part because the configuration had been selected on the evaluation data (Section 4.4).

The study has limits. First, only one clinical dataset was examined, so the findings do not extend to other datasets. Second, the confidence intervals resample the 768 instances; they do not include the variance from re-drawing the training data, and the three repeats reuse the same instances and are not independent. Third, 35 unadjusted intervals are reported, so isolated exclusions of zero should be read with caution. Fourth, the search was finite: 23 random encoder settings plus the original, and small grids for the standard models, with accuracy as the selection criterion. A larger search or a different criterion might change the ranking among the models. Fifth, the k-nearest-neighbour graph is built transductively over all instances, so held-out features, though never labels, are visible during training, and only median imputation and k-nearest-neighbour graphs were used. Finally, no calibration, threshold tuning or external validation was done.

6 Conclusions

This study asked whether graph attention improves weighted-polling classification on the Pima Indians Diabetes Dataset when the comparison is made fairly. In nested

cross-validation with a tuned encoder, the GATv2 classifier reached 75.22% accuracy and an AUROC of 0.808, against 75.35% and 0.822 for raw-feature weighted polling and 75.61% to 76.74% and 0.822 to 0.833 AUROC for the standard classifiers. The tuned encoder was no more accurate than the original one. Relative to raw-feature polling, accuracy differences larger than about two points are excluded in either direction, but a benefit of up to about 1.9 points cannot be excluded. Sensitivity was between 53% and 63% for all models. These results are limited to this single dataset and do not show that graph attention cannot help on other clinical data.

The earlier evaluation, which scored nodes used to train the encoder, gave an accuracy about 3.8 points above the nested estimate. In this setting, the encoder should therefore be retrained inside every fold with a loss restricted to training labels, preprocessing should use training rows only, and hyperparameters, including those of the encoder, should be selected in an inner loop. Evaluation on additional clinical datasets, with thresholds tuned for sensitivity, would give stronger evidence about when graph attention helps.

List of abbreviations. ELU: exponential linear unit; GAT: graph attention network; GATv2: graph attention network, version 2 (dynamic attention); GCN: graph convolutional network; GNN: graph neural network; k-NN: k-nearest neighbour; LOO: leave-one-out; SVM: support vector machine.

Declarations

Ethics approval and consent to participate. Not applicable. This study is a secondary analysis of a publicly available, de-identified dataset; no new data were collected from human participants.

Consent for publication. Not applicable. This study does not contain any individual person's identifiable data, clinical forms, or images.

Availability of data and materials. The Pima Indians Diabetes Dataset analyzed in this study is available in the GitHub repository cited in the text Brownlee (2019). The header row was added to the file used for the analyses; no values were changed. The code implementing the nested cross-validation, the original evaluation protocol and the label-shuffle control developed for this study is openly available under an MIT license at https://github.com/blackcontractor90/gatv2-weighted-polling-diabetes, and has been archived with a persistent identifier via Zenodo: https://doi.org/10.5281/zenodo.23466912.

Competing interests. The author declares that there are no competing interests.

Generative Artificial Intelligence Disclosure. Claude was used for writing and reviewing analysis code and for language editing. The author ran all analyses, verified the results, and is responsible for all content.

Funding. This research received no specific grant or financial backing from any funding agency in the public, commercial, or not-for-profit sectors. The work was carried out independently by the author.

Authors' contributions. Farid Morsidi: Conceptualization, Data curation, Formal analysis, Investigation, Methodology, Resources, Software, Validation, Visualization, Writing – original draft, Writing – review & editing.

Acknowledgements. Not applicable.

References

Brody, S., Alon, U., Yahav, E. (2022). How attentive are graph attention networks? *International conference on learning representations (ICLR).*

Brownlee, J. (2019). *Datasets: Pima Indians diabetes (GitHub repository).* https://github.com/jbrownlee/Datasets. (Accessed: 2026-08-14)

Chen, H., Farokhi, S., Bladen, K., Karimi, H., Moon, K.R. (2026). *Random-forest-induced graph neural networks for tabular learning.* arXiv preprint arXiv:2602.24224.

Defilippo, A., Veltri, P., Liò, P., Guzzi, P.H. (2024). Leveraging graph neural networks for supporting automatic triage of patients. *Scientific Reports*, *14*, , https://doi.org/10.1038/s41598-024-63376-2

Kapoor, S., & Narayanan, A. (2023). Leakage and the reproducibility crisis in machine-learning-based science. *Patterns*, *4*(9), 100804, https://doi.org/10.1016/j.patter.2023.100804

Karatsiolis, S., & Schizas, C.N. (2012). Region based support vector machine algorithm for medical diagnosis on Pima Indian diabetes dataset. *12th IEEE international conference on BioInformatics and BioEngineering (BIBE 2012)* (pp. 139–144). IEEE.

Koklu, M., & Ozkan, I.A. (2020). Multiclass classification of dry beans using computer vision and machine learning techniques. *Computers and Electronics in Agriculture*, *174*, 105507, https://doi.org/10.1016/j.compag.2020.105507

Lachi, D., Azabou, M., Arora, V., Dyer, E. (2024). *GraphFM: A scalable framework for multi-graph pretraining.* arXiv preprint arXiv:2407.11907.

Li, C.-T., Tsai, Y.-C., Chen, C.-Y., Liao, J.C. (2025). Graph neural networks for tabular data learning: a survey with taxonomy and directions. *ACM Computing Surveys*, *58*(1), , https://doi.org/10.1145/3744918

Morsidi, F. (2024). Hybrid learning for bottlenose dolphin motion approximation using weighted polling KNN and Bayes mechanism. *International Journal of Software Engineering and Computer Systems*, *10*(2), 173–186, https://doi.org/10.15282/ijsecs.10.2.2024.14.0132

Morsidi, F. (2026). *Hybrid GATv2 and weighted-polling KNN/naive Bayes classification for multiclass dry bean variety identification.* (Manuscript submitted to the International Journal of Software Engineering and Computer Systems)

Reza, M.S., Amin, R., Yasmin, R., Kulsum, W., Ruhi, S. (2024). Improving diabetes disease patients classification using stacking ensemble method with PIMA and local healthcare data. *Heliyon*, *10*(2), e24536, https://doi.org/10.1016/j.heliyon.2024.e24536

Smith, J.W., Everhart, J.E., Dickson, W.C., Knowler, W.C., Johannes, R.S. (1988). Using the ADAP learning algorithm to forecast the onset of diabetes mellitus. *Proceedings of the annual symposium on computer application in medical care* (pp. 261–265). Retrieved from https://pmc.ncbi.nlm.nih.gov/articles/PMC2245318/ (PubMed Central ID: PMC2245318)

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., Bengio, Y. (2018). Graph attention networks. *International conference on learning representations (ICLR).*