UNet-GAT Segmentation of Water Channel

1st Mahmut Osmanovic
Department of Computing
Jönköping School of Engineering.
Jönköping, Sweden
osma21nr@student.ju.se

2nd Isac Paulsson
Department of Computing
Jönköping School of Engineering.
Jönköping, Sweden
pais20dv@student.ju.se

***Abstract*—Accurate identification of water channels such as ditches and natural streams is essential for informed land and environmental management. This study presents a two-stage deep learning pipeline combining a UNet-based semantic segmentation model with a Graph Attention Network (GAT) for post-processing and refinement. Models were trained on slope-derived Digital Elevation Models using stratified 10-fold cross-validation. The UNet achieved an average F1-score of 0.75 for ditches and 0.39 for streams, while the GAT subtly improved these scores slightly to 0.76 and 0.41, respectively. A Bayesian paired t-test with a Region of Practical Equivalence (ROPE) of ±0.05 indicated no statistically meaningful performance difference between the models. Despite this, both approaches outperform prior baselines, although earlier work incorporating uncertainty quantification still outperforms our models on ditches while achieving equivalent performance on streams.**

***Index Terms*—Deep learning, Semantic Segmentation, LiDAR, Digital Elevation Model (DEM), Ditches, Natural Streams, UNet, Graph Attention Network (GAT), Bayesian T-test.**

I. Introduction

Throughout Sweden, natural streams and man made ditches play a crucial role in environmental management, agriculture and forestry. Yet current digital maps are lacking or unavailable for small water features (< 6 m wide); it has been reported that only 55% of streams and 9% of ditches are present [1]. Sweden has been mapped by Lantmäteriet using Aerial Laser Scanning (ALS/Lidar). This provides feature data in the form of point clouds which can be converted to Digital Elevation Models (DEM) and used for semantic segmentation tasks, such as mapping water features. Recent deep learning methods [2–4] utilize these features together with manually labeled maps of streams and ditches. They demonstrate the potential to increase the accuracy and availability of maps detailing small water features. However, these methods are currently not accurate enough to rely on given the differing regulations between streams and ditches [3].

Specifically, natural streams fall under stricter environmental protection acts, mandating buffer zones, limiting alterations, and safeguarding biodiversity. In contrast, man-made ditches, primarily constructed for drainage in agricultural or forestry contexts, are governed by regulations that emphasize their functional maintenance, but typically allow for more alteration. This distinction highlights the need for accurate and reliable mapping solutions that are able to differentiate between these two functionally and legally distinct water feature types to support compliant land management.

II. Related Work

Lidberg et al. [2] introduced a deep learning methodology for mapping drainage ditches in forested landscapes using airborne laser scanning (ALS) data. The dataset included 1607 km of digitized ditches and was used to train a Unet model for semantic segmentation. Their model achieved a recall of 86% with an MCC of 0.78 using only one topographical index (High Pass Median Filter).

Busarello et al. [3] Extended Lidbergs work by including additional topographical indecies and adding the prediction of streams. They found that the best performing topographical index for streams was the slope. In multi-class prediction they achieved an MCC of 0.63 for ditches and 0.28 for streams using slope as the only topographical index. Notably a major issue with the multi-class predictions is the propensity of the model to misclassify smaller segments of a stream or ditch as the opposite. Building on this, recent work by Westphal et al. [4] has focused on improving model reliability through uncertainty quantification. Monte Carlo dropout has been widely used to estimate predictive uncertainty, but often produces overly confident outputs. More recent approaches, such as conformal regression and Feature Conformal Prediction (FCP), offer calibrated uncertainty estimates and can highlight unreliable predictions. These methods have demonstrate potential in reducing misclassifications between streams and ditches, particularly in cases where features are narrow or ambiguous.

Nguyen and Thai [5] compared various loss functions and their impact on unbalanced datasets, they found that Tversky loss is highly efficient for heavily unbalanced data. It provided an increase in IoU scores from 0.329 with Binary Cross Entropy, to 0.374 with Tversky loss, on their most challenging dataset.

Kavran et al. [6] Utilized Graph neural networks to perform spatiotemporal semantic segmentation of land cover using multispectral satellite imagery. They utilized a super pixel approach with EfficientNetV2-S for feature extraction and a Graph-SAGE network for node classification. This enabled them to achieve an F1 score of 0.841, compared to the previous state of the art UNet's F1 score of 0.824.

Velickovic et al. [7] introduced Graph Attention Networks (GAT) which integrate an attention mechanism into a Graph Convolutional Network (GCN). By computing attention coefficients for each node's neighbors, GATs enable the model to dynamically weigh the importance of neighboring nodes during feature aggregation. Their approach is able to match or outperform state of the art models across several graph benchmark datasets.

Brody et al. [8] presented an improved variant of the original GAT that remedy limitations in the static attention mechanism of the original GAT. They propose a dynamic attention formula that allows for more expressive and flexible attention scores. Experimental results show that the new attention formula outperforms the original GAT implementation in various benchmarks.

III. Background

GATs [7] utilize attention mechanisms to aggregate information from neighboring nodes in a graph. The formula for aggregation is detailed in equation 1.

$$x'_i = \sum_{j \in \mathcal{N}(i) \cup \{i\}} \alpha_{i,j} W_{X_j}, \tag{1}$$

Where the attention weights $\alpha_{i,j}$ are calculated using equation 3, as introduced by Brody et al. [8]

$$e_{ij} = \mathbf{a}^{\top} \text{LeakyReLU}\left(W_s \mathbf{x} i + W_t \mathbf{x} j\right). \tag{2}$$

$$\alpha i, j = \frac{\exp(eij)}{\sum_{k \in \mathcal{N}(i) \cup i} \exp(e_{ik})}. \tag{3}$$

The attention mechanisms present in GATs provide added resilience against the oversmoothing problem commonly found in deeper conventional Graph Convolutional Networks.

$$\text{conv}(N_i, C_{\text{out}_j}) = \sum_{k=0}^{C_{\text{in}}-1} \text{weight}(C_{\text{out}_j}, k) \star \text{input}(N_i, k) \tag{4}$$

$$\text{out}(N_i, C_{\text{out}_j}) = \text{bias}(C_{\text{out}_j}) + \text{conv}(N_i, C_{\text{out}_j}) \tag{5}$$

Equations 5 and 4 describe the standard 2D convolutional operator, where $\star$ symbolizes the 2D cross-correlation operator. It functions by applying a set of learnable filters (kernels) to an input feature map. Each filter slides across the input, computing a dot product between its weights and the local region of the input it covers. This process extracts local patterns and features.

$$\mathcal{L}_{\text{Tversky}} = 1 - \frac{\text{TP}}{\text{TP} + \alpha\,\text{FP} + \beta\,\text{FN}} \tag{6}$$

Equation 6 details the Tversky Loss function, a metric-based loss particularly suited for image segmentation where class imbalance is present. By utilizing the parameters α and β, the loss function can be explicitly adjusted to impose a greater penalty on the missed detections of the minority class. The constants α and β were chosen based on previous research success. Specifically, constants $\alpha = 0.3$ and $\beta = 0.7$ were chosen [5].

IV. Data Exploration

For this article, we used a dataset provided by Busarello et al. [3]. The data chips themselves were processed from 12 distinct regions in Sweden. The resolution is 0.5 m, corresponding to 500x500 pixels representing areas of 250 m x 250 m.

The dataset consisted of 4593 slope images and a corresponding amount of annotated class labels. The annotated labels contain three classes: *background*, *ditch* and *stream* pixels. Whilst all labels contain the background class, that is not true of the remaining two classes. Approximately 92.27% of the labels include ditch pixels, while only 22.93% of the chips contain stream pixels. The mean frequency of pixel-wise class occurrence over all chips is highlighted in table I. It highlights one of the main obstacles in ascertaining a high performing semantic segmentation model on this task, namely the colossal class imbalance.

TABLE I
Mean Pixel Count and Frequency per Class

Class	Pixel Count	Frequency
Background	246,925	0.988
Ditch	3,029	0.012
Stream	1,223	0.005

K-fold cross validation was used as an evaluation technique. The distribution of chips in each fold was stratified based location. The specific quantity of class occurrence in the training and test sets for each fold is detailed in table II. Noteworthy is that the total chip count in each set always is equal to the background count (since each label contains at least one background pixel). Observe that the quantity of utilized chips per fold is significantly lesser than the total available chips ($\approx$ 3500 chips). The reason being that we strive to benchmark the M$_{UNet}$ and M$_{GAT}$ against previous state of the art research. In particular, the UNet with uncertainty quantification (UQ), developed by Westphal and others [4].

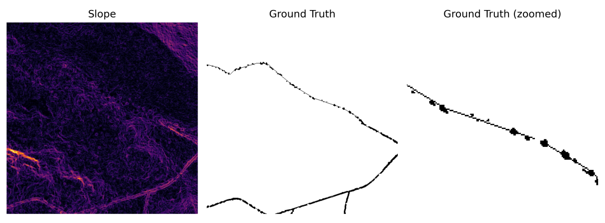

Fig. 1. Sample of slope and corresponding ground truth label, in this case only ditch and background

Notable in figure 1 is the occurrence of single pixel wide ditches. This is a consequence of how the image labels were created. The original labels consist of vector lines, which have been rasterized, widened and filtered by thresholding with a topographical index (HPMF) [3]. In order to not break connectivity after filtering, the rasterized 1 pixel wide vector lines have been added back to reform original connections.

TABLE II
FOLD-WISE CLASS DISTRIBUTION FOR TRAINING AND TEST SETS.

	Training Set			Test Set		
Fold	Background	Ditch	Stream	Background	Ditch	Stream
1	3044	2826	674	386	355	107
2	3039	2789	743	397	382	60
3	3045	2784	789	381	354	68
4	3060	2811	776	371	328	120
5	3088	2843	745	369	350	91
6	3116	2875	719	353	322	94
7	3084	2823	739	403	380	100
8	3082	2834	757	371	339	73
9	3036	2785	783	399	371	68
10	3047	2820	754	392	345	129
Mean	3064	2819	748	382	353	91
Ratio	1.00	0.92	0.24	1.00	0.92	0.24

TABLE III
AGGREGATED SEGMENT FEATURES

Feature Base	Aggregation	GAT Feature
node$_{id}$	identifier	No
center$_x$, center$_y$	coordinates	No
area	raw value	Yes
flow accumulation	min, sum, max, std	Yes
slope	min, mean, max, std	Yes
wetness index	min, mean, max, std	Yes
prob	mean$_0$, mean$_1$, mean$_2$	Yes
target	label	No

V. APPROACH DESCRIPTION

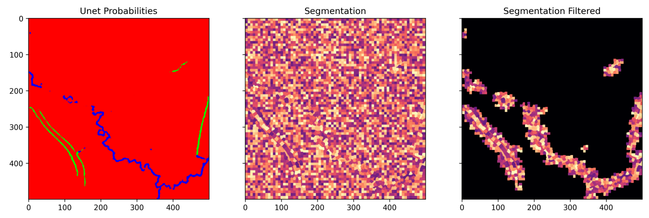

Fig. 2. Graph generation process. Colors for segments in the center and right image are randomized so that they can be differentiated from other segments.

Figure 2 illustrates the process of building graphs using a super-pixel based approach. Aggregating multiple pixels into larger segments (super-pixels) to form nodes results in a reduction of size and complexity in the graphs. The segmentation map is generated by applying the *SLIC* image segmentation algorithm to the predicted probabilities from M$_{UNet}$. This enables the graph and its segments to closely represent the predictions from M$_{UNet}$. This is appropriate since the main objective is to correct the predictions made by M$_{UNet}$. An additional benefit is that the segments formed in regions with low variation in probabilities become square. These segments are filtered out based on how well they fill their bounding box. By setting a threshold (0.99) on filtering, the area surrounding the channels are included in the graph. Edges in the graph are formed by connecting all immediately neighboring segments. Aggregated features are listed in table III. For the sake of comparison with previous predictions an image can be reconstructed from node predictions and the segmentation map.

The GAT based node classifier detailed in figure 3 was used to perform classification on the constructed graphs. The network begins with a linear projection of the input features, which is followed by graph normalization. The GAT core of the network consists of five layers with four attention heads per layer and a hidden dimension of 128, each layer is followed by a graph normalization layer. The final layer in the GAT core performs no concatenation and instead produces a unified feature vector per node. For classification, the node embeddings are passed through three linear layers progressively reducing the dimensionality to the final number of output classes, in our case 3. Five layers was selected to enable the model to capture relatively long distance relationships as discussed by Velickovic [7]. The training settings and hyperparameters are listed in table IV. In an effort to augment the feature set provided to the M$_{GAT}$, we experimented with extracting and compressing feature maps from various layers of the M$_{UNet}$ using dimensionality reduction (PCA). However, these additional features were not included in the final model since initial evaluations showed no improvement.

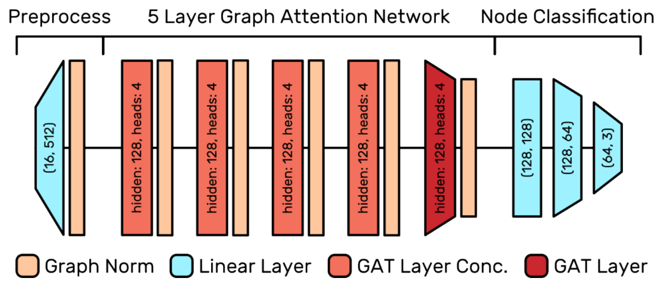

Fig. 3. Architecture of the proposed GAT-based model, featuring multiple encoder layers with 128 hidden units and 4 attention heads, followed by fully connected layers for output prediction.

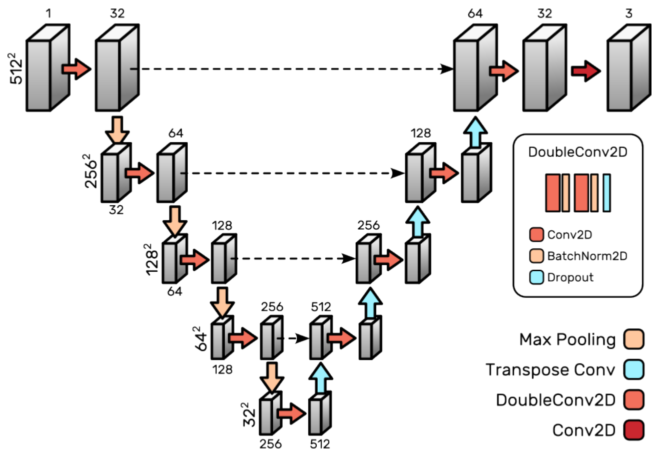

Fig. 4. An overview of the U-Net architecture used for semantic segmentation is presented. The model consists of a contracting path for hierarchical feature extraction and an expanding path for precise localization, with skip connections enabling the transfer of spatial information across corresponding resolution levels.

The UNet illustrated in figure 4 follows the previously established structure [2–4]. Inside the DoubleConv2D batch normalization is performed after each convolutional layer. Instead of using Weighted MSE with weights inferred by Median Frequency Balancing [4], Tversky loss is utilized.

TABLE IV
TRAINING SETTINGS AND HYPERPARAMETERS

GAT Parameter	**Value / Option**
Optimizer	Adam
Learning Rate	5×10^{-4}
Weight Decay	1×10^{-6}
Loss Function	Weighted CrossEntropyLoss
Class Weights	[1.0, 1.2, 2.0]
Dropout Rate	0.1
Batch Size	4
Epochs	15
UNet Parameter	
Optimizer	Adam
Learning Rate	8×10^{-4}
Loss Function	TverskyLoss
Tversky alpha	0,3
Tversky beta	0,7
Dropout Rate	5×10^{-2}
Batch Size	8
Epochs	100

$$Y_{final} = (\mathcal{M}_{GAT} \circ \mathcal{M}_{UNet})(X_{in}) \tag{7}$$

The final approach leverages a two-stage process (equation 7), combining the strengths of UNet for initial pixel-level segmentation and GAT for graph-based refinement. The UNet, with its convolutional architecture, excels at capturing local spatial features and provides a good foundation for building graphs using super-pixels. The GAT architecture through attention in multiple layers is able to capture long range dependencies in the constructed graphs, effectively correcting errors made by the UNet that rely primarily on the local features.

VI. Experiment Design

A. Hardware Requirements

Training and testing of the models were performed across two GPUs: an NVIDIA GeForce GTX 1080 Ti (11GB VRAM) and an NVIDIA A100 80GB. For the A100, only one-tenth of its capacity was utilized for both compute and VRAM. A notable difference in training duration was observed, with UNet-based models requiring approximately 6.5 hours across both devices, while the M$_{GAT}$-based models completed training in roughly 20 minutes. Additionally, UNet predictions necessitated a 10-minute preprocessing step, culminating in an effective total training time of around 70 hours.

B. Evaluation Technique

There were 20 experiments that were performed in total. Ten of these experiments were dedicated to train and test M_{UNet} models, while the other ten trained and tested M_{GAT} models. Each M_{UNet} model was trained and tested on a unique combination of chips. Specifically, the data was decomposed in different datasets through the use of $k - fold$ cross validation (where $k = 10$). Moreover, the two models were trained on the same $10 - fold$ cross validation splits. Cross validation is an evaluation technique that tests a trained model on all available data points exactly once. Specifically, the data is split up in ten chunks (in this case, stratified on location and class distributions), where each chunk is tested on once. Observe that each data point is used 9 times over upon training any of the two specific models, hence any difference in performance is not independent of each other [9].

C. Evaluation Metrics

The utilized evaluation metrics include precision (P), recall (R), the F1-score ($F1$), Intersection-over-Union (IoU) and Matthews Correlation Coefficient (MCC). Each experiment generated one set of class-values for each of the metrics. The fundamental metrics are precision and recall, from which the F1-score and IoU is derivable (not MCC).

More specifically,

$$F1(P, R) = 2\frac{P \times R}{P + R}, \tag{8}$$

where P is precision and R is Recall.

$$IoU(P, R) = \frac{P \times R}{P + R - P \times R}, \tag{9}$$

Let $D = (TP + FP)(TP + FN)(TN + FP)(TN + FN)$. The Matthews Correlation Coefficient (MCC) is then defined as:

$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{D}}, \tag{10}$$

where TP is true positives, TN is true negatives, FP is false positives and FN is false negatives.

These chosen metrics collectively provide a comprehensive evaluation of the models' performance, especially critical for tasks involving imbalanced classes like those often encountered in image segmentation. While precision and recall offer insights into false positives and false negatives respectively, the F1-score provides a single harmonic mean that balances these two, offering a robust measure of overall accuracy. Intersection-over-Union (IoU) is particularly relevant for segmentation tasks as it directly quantifies the overlap between predicted and ground truth segments. Finally, the Matthews Correlation Coefficient (MCC) stands out as a balanced metric that considers all four confusion matrix values (TP, TN, FP, FN). It's especially valuable because it produces a high score only if the classifier performs well on all of them, making it a reliable metric even when dealing with highly imbalanced datasets.

D. Procedure

The experiments were conducted in a two step pipeline. First, for a given fold, the M_{UNet} model was trained with the chips specified by that fold. For each input training image, the M_{UNet} generated a corresponding prediction image. In addition, the M_{UNet} model was evaluated on the test set found

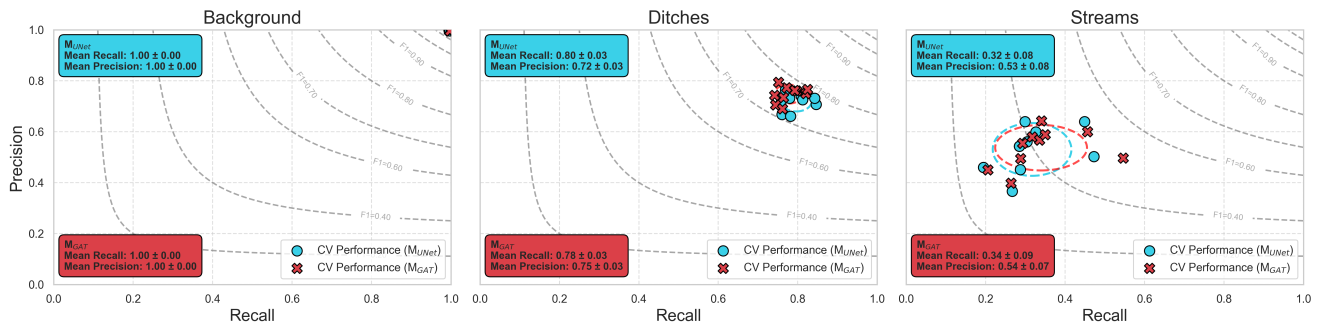

Fig. 5. Precision-Recall plots with F1-score isocurves for three semantic classes: *Background*, *Ditches*, and *Streams*. Each point represents a cross-validation fold. Circles indicate the performance of the baseline, model (M_{UNet}), while squares indicate the performance of the proposed model (M_{GAT}). F1-score contours in the background illustrate the trade-off between precision and recall. Mean precision and recall values (± standard deviation) are shown for each model and class.

within the fold in question. Thereafter, the M_{GAT} received a processed version of the prediction image (in addition to other features detailed in 1). It was subsequently trained and tested on the same datasets. The procedure was repeated for all folds.

E. Bayesian Hypothesis Testing

The performance (in terms of the detailed metrics) of each model on all test sets found within which fold was aggregated. Rather than merely testing for significance (e.g., students t-test) we chose to analyze the data through the lens of a *Bayesian t-test* [10]. This approach offers several crucial advantages over traditional null hypothesis significance tests (NHSTs), which are commonly used but have recognized limitations, particularly in the context of cross-validation. This approach offers several crucial advantages over traditional null hypothesis significance tests (NHSTs), which are commonly used but have recognized limitations, particularly in the context of cross-validation.

We chose a Bayesian t-test for correlated observations due to its suitability for our experimental design. Traditional t-tests assume independent observations, which doesn't apply here since both M_{UNet} and M_{GAT} models were evaluated on the exact same 10-fold cross-validation splits, resulting in correlated performance metrics. This Bayesian approach correctly handles such paired data, offering a statistically sound comparison. Furthermore, it directly answers our core question: the probability of a meaningful performance difference between the models, unlike traditional methods that only provide a p-value. Crucially, the Bayesian t-test allowed us to define a Region of Practical Equivalence (ROPE). We set this ROPE as an F1-score difference of 0.05, meaning we'd only consider one model significantly better or worse if its F1-score differed by at least that much. This ensures our conclusions about performance differences are not just statistically significant, but also practically meaningful.

VII. Result Analysis

Figure 5 highlights several findings. The first being that the performance boost from the M_{GAT} to the M_{UNet} model is marginal. Both models almost achieve a perfect score on the background pixels. As background pixels make up the majority of most images, they are also the simplest to predict. Although the ditches only make up about 1% of total image pixels (see table I), they do occur in 92% of training and test chips. Hence, predicting the ditches is a non-trivial task. The M_{UNet} model achieves a slightly larger recall than the M_{GAT} (about 80% vs 78%), though, at the expense of a worse precision (about 78% vs 75%). Nonetheless, the data points are concentrated, variation in performance is low and consistent across folds. In the stream class, we do not observe such performance consistency. Not only do streams make up a miniscule fraction of total pixels (as observed in I), but the quantity of chips with at least one stream pixel is merely a quarter of ditch pixels. On top of that, streams tend to be more stochastic in terms of their appearance, whilst ditches, as they are artifacts, often can be encapsulated by straight lines. Thus not only is the stream class infrequent but also complex, which shows in the stream model performance.

TABLE V
Class-wise MCC, IoU, and F1 for M_{UNet} across 10-fold cross-validation (mean ± std).

Class	MCC	IoU	F1
Ditches	0.7510 ± 0.0246	0.6066 ± 0.0307	0.7547 ± 0.0241
Streams	0.4084 ± 0.0663	0.2462 ± 0.0568	0.3918 ± 0.0720

TABLE VI
Class-wise MCC, IoU, and F1 for M_{GAT} across 10-fold cross-validation (mean ± std).

Class	MCC	IoU	F1
Ditches	0.7592 ± 0.0231	0.6156 ± 0.0305	0.7616 ± 0.0236
Streams	0.4232 ± 0.0703	0.2604 ± 0.0584	0.4098 ± 0.0737

The similarity in performance is not merely illustrated in figure 5, but confirmed in a Bayesian t-test (table VII). Given a Region of Practical Equivalence of ±0.05 (chosen due to low sample size and to enable comparison with previous research papers) in F1 score, almost the entirety of the area falls within

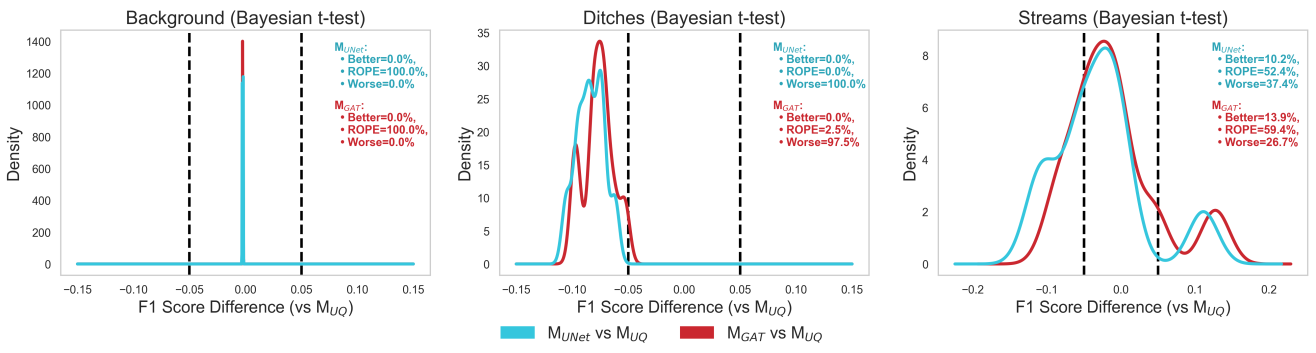

Fig. 6. Bayesian paired t-tests comparing the F1 scores of two models, M_{UNet} (blue) and M_{GAT} (red), against the baseline M_{UQ} across 10-fold cross-validation. Each subplot represents a semantic class: *Background*, *Ditches*, and *Streams*. The density plots illustrate the posterior distributions of F1 score differences relative to M_{UQ}, with vertical dashed lines indicating the Region of Practical Equivalence (ROPE), set at ± 0.05. Text annotations indicate the posterior probability that each model performs better than, worse than, or is practically equivalent to the baseline, allowing assessment of practical significance.

the ROPE. There may be several reason as to why that is. One possibility is the lack of meaningful features required to define the decision boundary between streams and ditches. Further aggregation of data from more topographical indices could improve the performance in classification by the M_{GAT}. Another possible reason could be that the graph construction is based on predicted probabilities from M_{UNet}. Hence, the M_{GAT} is unable to gain information from areas where M_{UNet} has been unable to find streams or ditches. While we do keep some of the surrounding area around predicted water features, inclusion of further areas may improve M_{GAT} performance. The method of graph construction also hinders the M_{GAT} from making predictions outside of correcting previous predictions, if a stream or ditch is missing from the M_{UNet} prediction the M_{GAT} won't be able to find it since it is not included in the graph.

As illustrated in Figure 1, there exist challenging regions within the label images. Specifically the single pixel wide features, these features exacerbate the problem of class imbalance and are challenging for M_{UNet} to predict. For performance evaluation they have the effect of weighting the features based on feature width, where thinner features have a lesser impact on the evaluation metrics. Alternative filtering methods ?should? be considered, with the goal of lessening class imbalance and to better represent the relevant features. This would also increase the accuracy of the evaluation metrics when compared to the original vector labels. These are issues introduced to M_{GAT} by the M_{UNet} processing. An alternative, fully graph based approach would be to build graphs from the base DEMs, with the goal of covering nodes and connections from the original vector labels. Hence, Providing a structural representation more suitable for graph neural networks (GNNs), without the issues of current pixel based approaches.

Y_{final} sub-models (M_{GAT} and M_{UNet}) were compared with previous state-of-the-art research. Both underperformed when compared to M_{UQ} by Westphal et al. [4]. In comparison with earlier work we see an uplift in performance. In particular, Busarello et al. achieve a maximum MCC of 63 and 28 on ditches and streams (slope index), both of which are lower than the scores achieved by any of our models under 10-fold cross-validation [3].

TABLE VII
BAYESIAN T-TEST RESULTS ON F1-SCORE DIFFERENCES: $\mathrm{M_{GAT}}$ VS $\mathrm{M_{UNet}}$

Category	$\mathbf{M_{GAT}}$ (%)	ROPE (%)	$\mathbf{M_{UNet}}$ (%)
Background	0.0	100.0	0.0
Ditches	0.0	100.0	0.0
Streams	3.5	96.5	0.0

Figure 6 shows that our models perform equivalently to M_{UQ} on background pixels. For the ditch class, $P(M_{UQ} \gg M_{UNet}) = 100\%$ and $P(M_{UQ} \gg M_{GAT}) = 97.5\%$, showing that M_{UQ} readily outperforms both M_{UNet} and M_{GAT}. For the streams class, $P(M_{UQ} \gg M_{UNet}) = 37.4\%$ and $P(M_{UQ} \gg M_{GAT}) = 26.7\%$, suggesting that while M_{UQ} may be better, the evidence is only weak to moderate. None of the models achieve a high enough posterior probability ($> 90\%$) in either the "Better" or "Worse" region, and most of the posterior mass lies within the ROPE or is split across categories. Therefore, we cannot conclude that either M_{GAT} or M_{UNet} is significantly better or worse than M_{UQ} in stream prediction.

VIII. Conclusion

This work introduces a two-stage deep learning framework that combines UNet-based semantic segmentation with a Graph Attention Network for the classification of ditches and natural streams using LiDAR-derived elevation data. Through 10-fold cross-validation, both models demonstrated strong and consistent performance, achieving F1-average scores up to 0.76 for ditches and 0.41 for streams, despite severe class imbalance. A Bayesian paired t-test confirmed that the observed performance differences between the models fall within the Region of Practical Equivalence, indicating that the GAT did not meaningfully improve performance. Nonetheless, the proposed models perform better or equivalent to previous methods that do not incorporate uncertainty quantification and perform comparably to state-of-the-art approaches that do, except on ditches.

References

[1] Jonatan Flyckt, Filip Andersson, Niklas Lavesson, Liselott Nilsson, and Anneli M. Å gren. Detecting ditches using supervised learning on high-resolution digital elevation models. *Expert Systems with Applications*, 201:116961, 2022.

[2] William Lidberg, Siddhartho Shekhar Paul, Florian Westphal, Kai Florian Richter, Niklas Lavesson, Raitis Melniks, Janis Ivanovs, Mariusz Ciesielski, Antti Leinonen, and Anneli M. Ågren. Mapping drainage ditches in forested landscapes using deep learning and aerial laser scanning. *Journal of Hydrologic Engineering*, 2023. Early Access.

[3] Mariana Dos Santos Toledo Busarello, Anneli M. Ågren, Florian Westphal, and William Lidberg. Automatic detection of ditches and natural streams from digital elevation models using deep learning. *Computers & Geosciences*, 176:105391, 2025.

[4] Florian Westphal, William Lidberg, Mariana Dos Santos Toledo Busarello, and Anneli M. Ågren. Uncertainty quantification for lidar-based maps of ditches and natural streams. *Environmental Modelling & Software*, 191:106488, 2025.

[5] Quang Du Nguyen and Huu-Tai Thai. Crack segmentation of imbalanced data: The role of loss functions. *Engineering Structures*, 297:116988, 2023.

[6] Domen Kavran, Domen Mongus, Borut Žalik, and Niko Lukač. Graph neural network-based method of spatiotemporal land cover mapping using satellite imagery. *Sensors*, 23(14), 2023.

[7] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *6th International Conference on Learning Representations (ICLR)*, Vancouver, Canada, 2018. Published as a conference paper at ICLR 2018.

[8] Shaked Brody, Uri Alon, and Eran Yahav. How attentive are graph attention networks?, 2022.

[9] A. Benavoli, G. Corani, J. Demšar, and M. Zaffalon. Time for a change: a tutorial for comparing multiple classifiers through bayesian analysis. *Journal of Machine Learning Research*, 18:1–36, aug 2017. Submitted 6/16; Revised 5/17; Published 8/17.

[10] Alessio Benavoli, Giorgio Corani, Francesca Mangili, and Marco Zaffalon. A bayesian nonparametric procedure for comparing algorithms. In Francis Bach and David Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 1264–1272, Lille, France, 07–09 Jul 2015. PMLR.