Ministry of Higher Education and Scientific Research
وزارة التعليم العالي والبحث العلمي

University of Badji Mokhtar
Annaba
Faculty of Technology
Electronic Department

جامعة باجي مختار عنابة
كلية التكنولوجيا
قسم الإلكترونيك

Detection et suivi de bords de routes dans les images d'UAV (Unmanned Aerial Vehicle)

Submitted to obtain the diploma of

Doctorat L.M.D

Field: Electronique
Specialization: Electronique

By

Rayene DOGHMANE

Thesis Supervisor
MCA. Karima BOUKARI U.B.M. - Annaba

Thesis Defense Committee
Chair: Prof. Mohamed Fezari U.B.M - Annaba
Examiner: MCA. Narima Zermi U.B.M - Annaba
Examiner: Prof. Ouchtati Salim UNIV - Skikda
Examiner: Prof. Abdelkrim Moussaoui UNIV - Guelma

Year: 2025/2026

"كشف وتتبع حواف الطرق في صور الطائرات بدون طيار"

الملخص

في السنوات الأخيرة، مع ظهور الرؤية الحاسوبية للتعلم العميق، أصبحت الشبكات العصبية الترشيحية (شبكات الالتفاف) CNNs أساسًا لمعالجة الصور. تتفوق هذه الشبكات في مهام مثل التعرف على الأشياء، التصنيف، والتقسيم. ومن بين هذه المهام، استفادت عملية استخراج الطرق من صور UAV بشكل كبير من CNNs بسبب قدرتها على تعلم الميزات المعقدة تلقائيًا من الصور. ومع ذلك، لكي تحقق CNNs أداءً مثاليًا، يجب أن يتوافق تصميم النموذج مع القيود الحسابية. تُعد U-Net واحدة من أشهر طرق لتعلم العميق لتقسيم الصور، وقد أظهرت قدرة عالية على التكيف وأداء متميز في الصور الجوية، مما يجعلها واعدة جدًا في اكتشاف الطرق.

في هذا المشروع البحثي، تم استخدام بنية محسنة لشبكة U-Net لاكتشاف الطرق تلقائيًا من صور UAV. تم استخدام مجموعة بيانات مفتوحة المصدر للطرق الجوية لتدريب النموذج. تم تحسين النموذج عبر دمج طبقة تقييس جديدة. كما تم استخدام تقنيات معالجة الصور وزيادة البيانات لتحسين الأداء. وأخيرًا، تم استخدام إحصائيات نوعية وكمية لتوضيح تفوق طريقتنا مقارنة بالنماذج الأساسية الأخرى.

أظهرت التجارب أن نموذج U-Net المحسن لدينا، المسمى BGN-UNet، حقق معدل تقاطع متوسط (Mean IoU) بلغ 0.984 بعد 25 دورة تدريبية فقط. تشير هذه النتيجة إلى تقارب أسرع وجودة تقسيم فائقة مقارنة بالنماذج الأساسية المختارة.

هذا المشروع البحثي هو عمل تم تنفيذه في مختبر LASA بجامعة باجي مختار عنابة. الهدف الرئيسي من هذا البحث هو تطوير منهجية للكشف التلقائي عن الطرق من صور UAV.

الكلمات المفتاحية: التجزئة الدلالية، U-Net، التقييس المجمّع الدفعي، صور من مركبة جوية بدون طيار، اكتشاف الطرق، التعلم العميق، تقنيات التقييس.

”Road Edges Detection and Tracking from UAV Images”

Abstract

In recent years, with the advent of computer vision and deep learning, Convolutional Neural Networks (CNNs) have become foundational for image processing. CNNs excel at tasks such as object recognition, classification, and segmentation. Among these tasks, road extraction from UAV imagery has benefited from CNNs due to their ability to automatically learn complex features from images. However, for CNNs to perform optimally, the model architecture must align with computational constraints. U-Net is one of the most well-known deep learning methods for image segmentation and has demonstrated remarkable adaptability and performance on aerial images. It shows great promise for road detection.

In this thesis project,an enhanced U-Net architecture is used to automatically detect road from UAV images. An opensource aerial road dataset is used for training the model. The model is improved by integrating new normalization layer. Image pre-processing techniques and augmentation methods are also used for improvement. At last, qualitative and quantitative statistics are used to illustrate superiority of our method compared to other baseline models.

The experiments demonstrate that our enhanced U-Net model, called BGN-UNet, achieved a Mean Intersection over Union (Mean IoU) score of 0.984 after just 25 epochs. This result indicates both faster convergence and superior segmentation quality compared to the selected baseline models.

This thesis project is a research project carried out by the LASA Laboratory at Badji Mokhtar University of Annaba. The main objective of this thesis is to develop a methodology for automatic road detection from UAV images.

Keywords: Semantic segmentation, U-Net, Batch Group Normalization, UAV imagery, road detection, deep learning, normalization techniques.

”Detection et Suivi de Bords de Routes dans les Images d’UAV”

Résumé

Ces dernières années, avec l’avènement de la vision par ordinateur et de l’apprentissage profond, les réseaux de neurones convolutionnels sont devenus fondamentaux pour le traitement d’images. Ces réseaux excellent dans des tâches telles que la reconnaissance d’objets, la classification et la segmentation. Parmi ces tâches, l’extraction des routes à partir d’images de drones a bénéficié des réseaux de neurones convolutionnels grâce à leur capacité à apprendre automatiquement des caractéristiques complexes à partir des images. Cependant, pour que ces réseaux fonctionnent de manière optimale, l’architecture du modèle doit être adaptée aux contraintes computationnelles. U-Net est l’une des méthodes d’apprentissage profond les plus connues pour la segmentation d’images et a démontré une remarquable adaptabilité et performance sur les images aériennes. Elle montre un grand potentiel pour la détection des routes.

Dans ce projet de thèse, une architecture améliorée de U-Net est utilisée pour détecter automatiquement les routes à partir d’images de drones. Des images aériennes issues d’une source ouverte, ont été utilisées pour entraîner le modèle. Ce dernier est amélioré par l’intégration d’une nouvelle couche de normalisation. Des techniques de prétraitement d’images et des méthodes d’augmentation de données sont également utilisées pour améliorer les performances. Enfin, des statistiques qualitatives et quantitatives sont employées pour illustrer la supériorité de notre méthode par rapport à d’autres modèles de référence.

Les expériences montrent que notre modèle amélioré U-Net, appelé BGN-UNet, a atteint un score moyen d’intersection sur l’union de 0,984 après seulement 25 époques. Ce résultat indique une convergence plus rapide et une qualité de segmentation supérieure par rapport aux modèles de référence sélectionnés.

Ce projet de thèse est une recherche réalisée au laboratoire LASA de l’université Badji Mokhtar d’Annaba. L’objectif principal de cette thèse est de développer une méthodologie pour la détection automatique des routes à partir d’images de drones.

Mots-clés : Segmentation sémantique, U-Net, Batch Group Normalization, images UAV, détection de routes, apprentissage profond, techniques de normalisation.

This thesis is dedicated to my mother

You are the light behind every word I wrote,
your absence lives in the spaces between them.
Though your hands cannot hold these pages,
your angelic heart guides every one.

Rayene,
your daughter who misses you endlessly

Acknowledgments

First and foremost, I extend my deepest gratitude to my supervisor, Dr. Karima BOUKARI, for her exceptional guidance and kindness throughout my PhD journey. Her unwavering support, patience, and insightful feedback were invaluable to this work. I am profoundly grateful for her mentorship, which was always delivered with generosity and respect—making our collaboration not only productive but also a deeply rewarding experience.

I would also like to thank my thesis committee members, for their time, thoughtful critiques, and constructive discussions, which helped shape this research.

To **my mother**, who is no longer here to witness this moment—your love was my first and most enduring strength. Every step I take is still guided by your wisdom. Though my heart aches that you cannot see this day, I find comfort in knowing you are under Allah's mercy, resting in Jannah's eternal peace. May the Most Merciful reunite us in His perfect time, and may my work continue to be a sadaqah jariyah that reaches you.

To **my grandfather**, whose absence is deeply felt—your quiet lessons and unwavering belief in me live on in all I do. I carry your legacy with pride, and I hope you rest in a peace this world could never offer.

To my father, my brothers and sisters, and my grandmother—thank you for your endless support and for standing by me throughout this journey. To my beloved nephews, your bright spirits reminded me of life's lightness even on the hardest days.

To my husband Imad, my rock and my greatest source of strength—your unwavering belief in me, your patience, and your love made this possible. No words can fully express my gratitude.

To my son, my brightest joy—thank you for inspiring me every day and for reminding me what truly matters. I hope this achievement makes you proud.

To my dearest friend, Nour—thank you for your steadfast support, and for being my anchor through both storms and sunshine. This journey would have been far lonelier without you.

Finally, to all those who have supported me in ways big and small—thank you.

Contents

1 General Introduction **1**

1.1 Motivation 2
1.2 Research Problems 3
1.3 Project Background 4
1.4 Research Objectives 5
1.5 Proposed Solutions 5
1.6 Research Scope 6
1.7 Structure of the Thesis 7

2 Literature Review **8**

2.1 Introduction 9
2.2 Traditional Approaches 9
2.3 Deep Learning for Road Extraction 10
2.4 Image Segmentation 10
2.5 Recent Architectures and Improvements 12
2.6 Representative UAV Road Detection Algorithms (Pre-2015 to 2025) . . . 13
2.7 Normalization Techniques 16
2.8 Research Gaps and Motivation 19

3 Methodology **22**

3.1 Introductinon 23
3.2 Memory Requirements to Train a Deep Learning Model 23
3.3 Dataset 26
3.4 Data Preprocessing 26
3.5 Data Augmentation 29
3.6 Normalization Techniques 30
3.7 Baseline Model 33
3.8 Proposed Model Architecture 33
3.9 Implementation and Training Setup 35
3.10 Evaluation Metrics 36
3.11 Summary 36

4 Experimental Results and Discussion **39**
4.1 Introduction . . . 40
4.2 Experimental Setup . . . 40
4.3 Quantitative Results . . . 40
4.3.1 Mean Intersection over Union (IoU) Analysis . . . 40
4.3.2 Inference Speed and Model Initialization . . . 41
4.3.3 Visual Evaluation . . . 42
4.4 Ablation Study: Impact of Batch Size . . . 43
4.5 Practical Applications of BGN-UNet in Road Detection . . . 48
4.6 Summary . . . 49

5 Challenges and Considerations for Practical Implementation of BGN-UNet **50**
5.1 Introduction . . . 51
5.2 Challenges Related to Normalization Techniques . . . 51
5.3 Implementation Considerations . . . 52
5.4 Training Considerations . . . 53
5.5 Computational and Memory Constraints . . . 53
5.6 Robustness and Generalization . . . 53
5.7 Future Directions and Recommendations . . . 54
5.8 Summary . . . 54

Appendix **57**
A.1 Classical Image Segmentation Methods . . . 58
A.2 BGN-UNet Model Building Algorithm . . . 59
A.3 Prediction with Trained BGN-UNet . . . 60

Bibliography **72**

List of Figures

1.1 Binary Semantic Segmentation for Aerial Road Image 3
1.2 Normalization Techniques: Batch Normalization (BN), Layer Normalization (LN), Group Normalization (GN), and Batch Group Normalization (BGN). The purple regions indicate the pixels used to compute normalization statistics. 6

2.1 Comparison of classical segmentation methods applied to UAV imagery: (a) Original image; (b) Global thresholding; (c) Canny edge detection; (d) Otsu thresholding; (e) Watershed segmentation; (f) K-means clustering. These classical methods often struggle with fragmented edges, noise, and over-segmentation due to shadows and complex backgrounds. 12
2.2 (a) Test accuracy of a neural network trained on MNIST with and without Batch Normalization (BN) versus training steps. BN accelerates training and improves accuracy. (b, c) Evolution of input distributions to a typical sigmoid activation during training, shown at the 15th, 50th, and 85th percentiles. BN stabilizes these distributions, reducing internal covariate shift. Figure adapted from Ioffe and Szegedy (2015) [17]. 17
2.3 DRAW model test negative log likelihood with and without layer normalization [22]. 18
2.4 Visualization of different normalization techniques in deep learning, including Batch Normalization, Instance Normalization, Layer Normalization, Group Normalization, Batch Group Normalization, and Switchable Normalization. Each colored region illustrates the scope over which statistics are computed for each normalization method. 19

3.1 Examples of training images and their corresponding ground truth masks. 26
3.2 Creating Patches. (a): Patches from padded graysclae image, (b): Patches from padded groundtruth 28
3.3 Elimination of Non-Informative Patches. (a):Informative images patches, (b): Informative groundtruth patches 28

3.4 Visualization of the preprocessing pipeline applied to the dataset images. The process begins with the input RGB image and its ground truth mask, followed by grayscale conversion, padding, patch creation, and pruning of non-informative patches. Each step is illustrated for both the image and its corresponding mask, providing a clear overview of the transformations performed prior to model training. 29
3.5 Example of data augmentation applied to an aerial image. The original image (bottom left) is shown alongside several augmented versions generated through flipping and rotation. These transformations increase the diversity of the training data and improve model robustness. 30
3.6 The original U-Net architecture illustrating the encoder-decoder structure with skip connections, which forms the baseline model used in this study [11]. 34
3.7 Comparison of convolutional blocks in the standard U-Net and the proposed BGN-UNet, highlighting the incorporation of Batch Group Normalization layers in the latter. 35
3.8 Architecture of the proposed BGN-UNet model, based on the original U-Net, with Batch Group Normalization integrated to improve segmentation performance and training stability. 36
3.9 Flowchart of the Proposed Methodology of the BGN-UNet approach for UAV road detection. 38

4.1 Inference times per patch for BGN-UNet on a 1024×512 image divided into 8 patches of 256×256 pixels. The first patch requires approximately 18 seconds due to initialization overhead, while subsequent patches are processed in about 125–132 milliseconds each. 42
4.2 Training and Validation Metrics for Batch Size 2: Loss and Accuracy for BN-UNet, GN-UNet, and BGN-UNet. 44
4.3 Training and Validation Metrics for Batch Size 8: Loss and Accuracy for BN-UNet, GN-UNet, and BGN-UNet. 45
4.4 Training and Validation Metrics for Batch Size 16: Loss and Accuracy for BN-UNet, GN-UNet, and BGN-UNet. 46
4.5 Road segmentation using BGN-UNet 47
4.6 Semantic Segmentation of the Aerial Road Image with BGN-UNet 48
4.7 Practical applications of the BGN-UNet model in UAV-based road detection. 49

List of Tables

2.1 Representative UAV Road Detection Algorithms Developed Between 2015 and 2025 16
2.2 Comparison of Normalization Techniques in Deep Neural Networks . . . 20

3.1 Model Hyperparameters 37

4.1 Mean IoU for Different Batch Sizes and Models 41
4.2 Initial Inference Time per Patch for Different Models 42

List of Acronyms

ASPP Atrous Spatial Pyramid Pooling

BGN Batch Group Normalization

BN Batch Normalization

CBAM Convolutional Block Attention Module

CEDL Cross-Entropy-Dice Loss

CGAN Conditional Generative Adversarial Network

CNN Convolutional Neural Network

DCGAN Deep Convolutional Generative Adversarial Network

Deep TEC Deep Transfer learning with Ensemble Classifier

DRAW Deep Recurrent Attention Writer

FBN Federated Batch Normalization

FCN Fully Convolutional Network

GAN Generative Adversarial Network

GPU Graphics Processing Unit

GN Group Normalization

HeMoDU High-Efficiency Multi-Object Detection Algorithm for Unmanned Aerial Vehicles

LN Layer Normalization

MRF Markov Random Field

MS-AGAN Multi-Scale Attention Generative Adversarial Network

ReLU Rectified Linear Unit

RCNN Region-based Convolutional Neural Network

RGB Red, Green, and Blue

SVM Support Vector Machine

UAV Unmanned Aerial Vehicle

List of Algorithms

1 Classical Image Segmentation Methods for UAV Imagery 58
2 Construction of the BGN-UNet Model 59
3 Patch-wise Prediction with Trained BGN-UNet 60

Chapter 1

General Introduction

Contents

1.1 Motivation 2
1.2 Research Problems 3
1.3 Project Background 4
1.4 Research Objectives 5
1.5 Proposed Solutions 5
1.6 Research Scope 6
1.7 Structure of the Thesis 7

1.1 Motivation

This thesis presents a framework for detecting roads in imagery captured by Unmanned Aerial Vehicles (UAVs). This task is essential for many applications, including autonomous navigation, infrastructure monitoring, urban planning, and disaster response [1][2]. Figure 1.1 presents an example of an aerial road image along with its corresponding binary segmentation mask, illustrating the semantic segmentation technique we used in this thesis. With the rapid growth of Unmanned Aerial Vehicle (UAV) technology, it has become possible to collect high-resolution aerial images at scales and frequencies that were not achievable before, creating new opportunities for automated road analysis. Despite these advantages, natural scene complexity, non-road objects, occlusions, and variations in lighting conditions remain major obstacles to the precise extraction of road boundaries from UAV imagery [3][4]. Relying on traditional manual inspection is no longer practical, as it is time-consuming, labor-intensive, and prone to error, which highlights the need for robust and fully automated solutions [5].

In order to address these challenges, we place our work within the broader field of computer vision, where image processing techniques have long been used to analyze and interpret visual data. Tasks such as edge detection, segmentation, and feature extraction were traditionally carried out using handcrafted algorithms [6][7][8]. However, when we apply these methods to UAV imagery, they often fail because of the high variability and complexity of aerial scenes [9]. For this reason, we turn to learning-based methods, where models are trained on large collections of images and learn directly from data instead of depending on manually designed rules.

Modern approaches like Convolutional Neural Networks (Convolutional Neural Network (CNN)s) have completely transformed computer vision by making it possible to learn features hierarchically and achieve impressive results in both image classification and segmentation tasks [10]. CNNs are particularly effective for road detection, where identifying fine details and larger structures is equally important, because they can automatically capture spatial patterns at different scales.

Based on this progress, one of the most widely used CNN architectures for segmentation is U-Net. The original U-Net was first introduced for biomedical image analysis. Its encoder–decoder design and skip connections allow the model to preserve spatial details while also capturing high-level semantic information. These two properties are essential for accurately detecting roads in UAV images [11]. However, even though U-Net provides a strong starting point, its performance can be affected by the choice of normalization technique, particularly when training with small batch sizes [12]. Working with small batches (which define the number of samples that are propagated through the network) is often unavoidable in aerial road segmentation, since high-resolution images quickly consume memory and force us to reduce the batch size to fit both the data and the model on GPUs

Figure 1.1: Binary Semantic Segmentation for Aerial Road Image

with limited resources [13].

1.2 Research Problems

Despite the progress in computer vision and deep learning, we still face many challenges when trying to accurately detect and track road edges in UAV images. One big difficulty comes from the fact that roads often look very similar to the surrounding areas, like soil, rooftops, or vegetation. On top of that, shadows, trees or buildings blocking the view, and changes in lighting or seasons make it even harder for us to get precise road edges, which can sometimes end up fragmented or unclear.

We also face challenges from the complex backgrounds in UAV images, especially in urban or semi-urban areas. These scenes are full of things like vehicles, pedestrians, buildings, and vegetation, which makes it tricky for us to separate roads from everything else. To deal with this clutter, our models need to be both accurate and aware of the surrounding context.

High-resolution aerial images add another layer of difficulty during training and deployment. Processing such large images uses a lot of GPU memory, which often forces us to work with smaller batch sizes to avoid running out of memory. Many popular models, including U-Net, use normalization layers like Batch Normalization, and these layers don't work well when the batch size is small. This can make training unstable and reduce the performance of our models.

On top of that, UAV platforms usually have limited computational resources. Deep learning requires heavy calculations and lots of processing power, but embedded UAV systems often have restrictions on memory, storage, and computing capacity. This makes it tough for us to deploy large models in real time or on a bigger scale.

Finally, although several advanced normalization techniques have been proposed to overcome the limits of traditional methods, we noticed that many of them have not been tested in U-Net for high-resolution aerial road segmentation. Their potential benefits like more stable training, faster learning, and better results with small batches are still underexplored. This is why we want to evaluate and adapt these techniques to see how well they

work for UAV-based road detection.

1.3 Project Background

Semantic segmentation has become a key task in computer vision, especially in applications that require a detailed understanding of scenes, such as medical imaging, autonomous driving, and aerial mapping [14]. In our work on UAV-based road detection, semantic segmentation helps us classify every pixel in an aerial image, allowing us to separate road surfaces from the background with high spatial accuracy.

The rise of deep learning, particularly Convolutional Neural Networks (CNNs), has greatly improved what we can do with semantic segmentation. CNNs let our models learn hierarchical features directly from image data, which makes them much more effective than traditional methods that rely on handcrafted features. This ability is especially useful for handling the complexity of aerial imagery captured by UAVs.

Among CNN architectures, U-Net has become very popular because of its encoder-decoder design [11]. This structure allows us to capture both local textures and global semantic information efficiently. Although U-Net was originally developed for biomedical image segmentation, we and other researchers have adapted it for remote sensing and aerial image analysis because its skip connections help preserve spatial details between encoding and decoding paths [15, 16].

Even though U-Net provides a strong starting point, we know that its baseline implementation has some limitations. Its performance depends on factors such as the choice of normalization technique [17], the resolution of the training data, and how efficiently the model runs especially in environments with limited hardware. To overcome these issues, recent work has focused on enhancing U-Net with features like multi-scale context aggregation, attention modules [18], and alternative normalization layers, all aimed at improving accuracy and generalization in complex scenarios.

Despite these improvements, we noticed that many of them have not been fully tested under the specific constraints of UAV-based road detection, such as small batch sizes, high-resolution inputs, and the need for real-time deployment. This shows that there is still a need to adapt segmentation models carefully to balance performance, efficiency, and robustness in aerial road segmentation tasks.

Building on this background, in our research we aim to explore how U-Net can be effectively optimized for road detection in UAV imagery, by adjusting its components to better address these practical challenges.

1.4 Research Objectives

In this thesis, our main goal is to create and assess a strong deep learning framework that can detect roads in UAV imagery. Therefore, we aim to overcome persistent limitations in existing segmentation models, particularly those related to normalization techniques when applied under small batch training conditions [19]. As discussed earlier, high-resolution aerial images often impose memory constraints that necessitate the use of small batch sizes, making standard normalization methods like Batch Normalization unstable and less effective.

To address these constraints, we propose to integrate a more adaptive normalization strategy into the popular U-Net model to create a novel architecture that we named BGN-UNet. Our main objective is to systematically assess the impact of this modification on segmentation performance under controlled conditions that reflect common limitations in aerial image analysis, such as limited memory resources and hardware constraints. Additionally, our study seeks to analyze the practical considerations involved in using such models for UAV tasks, including computational efficiency and model scalability, while recognizing that real-world deployment remains a topic for future investigation.

To summarize the research objectives of this work, this thesis focuses on enhancing the baseline U-Net architecture through improved normalization, performing comparative analysis with existing approaches, and discussing the practical aspects of model adaptation in the context of UAV imagery.

1.5 Proposed Solutions

To address the challenges in road segmentation from UAV images, we propose an enhanced U-Net architecture called BGN-UNet that integrates Batch Group Normalization (BGN). It is important to note that the original U-Net architecture, as introduced for biomedical image segmentation, did not include any normalization blocks. While many subsequent implementations incorporated Batch Normalization to stabilize training and improve convergence, such additions are not standardized and often struggle under small batch size regimes common in high-resolution aerial image processing.

Batch Group Normalization (BGN) combines the strengths of Batch Normalization [20] and Group Normalization [21], making it more adaptable to varying batch sizes and feature structures. Figure 1.2 illustrates several normalization techniques, Batch Normalization (BN), Layer Normalization (LN) [22], Group Normalization (GN) and Batch Group Normalization (BGN) [23]. Each subfigure displays a feature map with axes representing batch size (N), number of channels (C), and spatial dimensions (H, W). The highlighted regions indicate the pixels used to compute normalization statistics. Unlike other methods, BGN uniquely merges the channel and spatial dimensions into unified

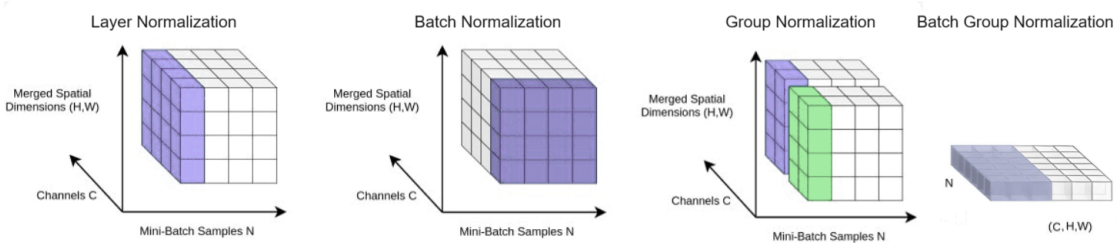


Figure 1.2: Normalization Techniques: Batch Normalization (BN), Layer Normalization (LN), Group Normalization (GN), and Batch Group Normalization (BGN). The purple regions indicate the pixels used to compute normalization statistics.

groups, enabling more stable and effective normalization, particularly when training with small batches.

The BGN-UNet [24] model is implemented and evaluated under realistic constraints associated with aerial imagery, such as processing high-resolution inputs under limited memory resources. Its performance is compared to the baseline U-Net variants employing traditional normalization strategies, with evaluation metrics focused on segmentation quality, training stability, and robustness. Although the experiments were conducted in a controlled research environment using cloud-based GPU resources. We found that using the right normalization method is key to improving a model's ability to deal with UAV segmentation images.

1.6 Research Scope

This thesis is centered on the problem of binary semantic segmentation, specifically, the detection and delineation of road pixels from non-road regions in UAV captured imagery. Our research is limited to experiments designed to simulate the constraints encountered in UAV image processing, such as large image dimensions, restricted computational resources, and the need to train with small batch sizes due to memory limitations.

The research scope includes the design, implementation, and evaluation of the proposed BGN-UNet model, with a focus on segmentation accuracy, training stability, and robustness to aerial scene content. A comparative analysis is conducted against traditional normalization methods and established segmentation architectures to determine the added value of BGN within the U-Net framework.

To manage computational limitations and focus the research on its main objective, we adopt a binary semantic segmentation approach, distinguishing only between road and non-road classes. This simplification is justified by the specific needs of aerial road extraction, where multi-class segmentation is unnecessary. Additionally, to reduce memory consumption and enhance training efficiency, grayscale images were used instead of Red, Green, and Blue (RGB), as color information was not essential for this task. High-

resolution aerial images were also divided into smaller patches to make training feasible on limited hardware resources. These choices, while detailed further in the Methodology chapter, define the practical and technical boundaries within which the BGN-UNet model was designed and evaluated.

Although our study does not involve deployment in real-world UAV systems, it includes a theoretical discussion of practical considerations such as model scalability and computational feasibility. These aspects are considered with the intention of guiding future research toward real-time or embedded applications.

1.7 Structure of the Thesis

The remainder of this thesis is organized into five chapters. In Chapter 2, we present a comprehensive literature review, covering traditional and deep learning-based approaches to road extraction, recent enhancements to the U-Net architecture, and normalization techniques that aim to improve training under constrained conditions. Chapter 3 outlines our research methodology, where we describe the experimental framework, including our dataset, model architecture, training procedures, and evaluation metrics, all adapted to the constraints of UAV-based segmentation. Chapter 4 presents our experimental results, offering quantitative and qualitative evaluations of our proposed BGN-UNet, alongside ablation studies and comparisons against baseline models. In Chapter 5, we discuss the challenges and theoretical considerations for practical implementation, focusing on model scalability, memory efficiency, and potential deployment constraints. Finally, the general conclusion summarizes our main contributions, identifies limitations, and outlines our vision for future research.

Chapter 2

Literature Review

Contents

2.1 Introduction 9
2.2 Traditional Approaches 9
2.3 Deep Learning for Road Extraction 10
2.4 Image Segmentation 10
2.5 Recent Architectures and Improvements 12
2.6 Representative UAV Road Detection Algorithms (Pre-2015 to 2025) . 13
2.7 Normalization Techniques 16
2.8 Research Gaps and Motivation 19

2.1 Introduction

Automated road extraction from aerial imagery, especially from Unmanned Aerial Vehicles (UAVs), has become a major focus in remote sensing and computer vision over the last ten years. This shift began moving away from older photogrammetric and manual methods [25], which, while useful for small mapping projects, were not easily scaled or automated. The rise of high-resolution sensors on flexible UAV platforms greatly improved data collection, paving the way for algorithmic solutions suited for dynamic and large-scale road detection [26, 27]. UAVs provide distinct benefits compared to satellites or manned aircraft, such as reduced cost, more adaptable deployment, and superior spatial and temporal detail [28]. These characteristics position UAV imagery as a highly suitable resource for applications like urban planning, traffic analysis, disaster management, and environmental monitoring [29].

Despite advances in the field, the extraction of road networks from UAV imagery still encounters many challenges. Roads vary considerably in their characteristics, with differences in width, construction material (asphalt, concrete, or gravel), surface condition, and spectral properties [30]. They are also frequently obscured by trees, vehicles, or buildings, and their appearance may be distorted by shadows, illumination changes, or oblique viewing angles [31]. Background features with similar visual patterns, such as rooftops, riverbanks, or parking areas, can easily be mistaken for roads, particularly in complex urban and rural environments [26]. These difficulties undermine the assumptions of earlier rule-based or feature-driven methods, which often suffer from limited generalization, high rates of false identification, and weak boundary precision [30, 31].

2.2 Traditional Approaches

Early methods for road extraction were mainly based on classical machine learning and hand-crafted features. In these approaches, techniques such as Support Vector Machine (SVM) [32], Markov Random Field (MRF) [33], edge detection, and template matching were commonly used. These approaches tried to recognize road pixels by using predefined cues such as geometry, texture, or spectral properties [25]. For example, Mie et al. combined SVM with template matching to make road detection more reliable [34], while Jayaseeli and Mathali applied multi-thresholding along with heuristic optimization to segment high-resolution satellite images [35]. While these methods managed to produce reasonable results, they often came with limitations. They required significant pre-processing and careful adjustment of parameters for each dataset, and they struggled in difficult situations such as road occlusion, varying illumination, and the inherent complexity of aerial imagery.

2.3 Deep Learning for Road Extraction

Deep learning approaches were developed to overcome the limitations of traditional methods and to improve both robustness and scalability. One important development was the Fully Convolutional Network (FCN) [1]. This network enabled pixel-wise prediction by removing the fully connected layers. However, because it relied on repeated downsampling, the results were often coarse and the boundaries of roads were not well defined.

Thus, encoder–decoder architectures were introduced to solve the problem of imprecise results. Examples of these are U-Net, SegNet [36], and DeConvNet [37]. These models use a symmetric design with skip connections or index-preserving upsampling to keep the spatial details and recover clearer segmentation maps. Among them, U-Net became the most popular because it is effective and can be adapted to many applications. Later versions such as Residual U-Net [38] added residual blocks to make training more stable. Buslaev et al. further improved the design by using ResNet encoders, which increased feature depth and made the extracted information richer [39, 40].

Further progress was achieved with dilated convolutions and multi-scale fusion methods. The DeepLab series followed this direction by adding the Atrous Spatial Pyramid Pooling (ASPP) module [41], which gathers contextual information at different scales without losing spatial detail. At the same time, attention mechanisms were introduced to emphasize important regions and reduce background noise, which improved both recall and precision [42]. Other models such as RoadNet and CasNet used multi-task learning, where surface segmentation, edge detection, and centerline extraction were performed together. This gave the models a stronger structural understanding of roads [43].

Even with these deep learning methods, the task of segmenting roads in UAV images is still challenging. The main difficulties are related to scale changes, occlusions, and similarities with other objects. The next section looks more closely at image segmentation methods, focusing on binary semantic segmentation and the architectural and normalization strategies used in our proposed approach.

2.4 Image Segmentation

Image segmentation is an important task in computer vision. It divides an image into meaningful parts that represent real-world objects or boundaries [14]. In this thesis, the focus is on **binary semantic segmentation**, where each pixel is labeled as road or non-road. This is essential in UAV-based road extraction, since segmentation helps isolate road areas from complex scenes. Such separation is important for applications like autonomous navigation, infrastructure monitoring, and post-disaster analysis.

Classical segmentation methods are usually divided into two groups: methods based on discontinuity and methods based on similarity. Discontinuity-based methods, such as

edge detection, work by finding sudden changes in pixel values. They can detect clear boundaries, but they are very sensitive to noise and often create broken edges, especially in aerial images affected by shadows or occlusions [44, 45, 46, 47, 48].

Methods based on similarity include thresholding, region growing, and clustering. Thresholding is simple and fast, but it cannot handle changes in lighting that are common in UAV images. Region growing is more robust to noise but depends on careful parameter choice and initialization. Clustering methods, such as k-means and fuzzy c-means, group pixels with similar features, but they often ignore spatial context. As a result, they may confuse roads with non-road areas when their spectral values overlap [49, 50].

Other classical methods include watershed algorithms, which treat grayscale images as topographic surfaces and segment them by finding catchment basins. These can produce closed boundaries but often lead to over-segmentation and require preprocessing [51]. PDE-based methods use diffusion to enhance structures, but they are computationally expensive and depend strongly on the starting conditions.

Even with these techniques, traditional methods face clear limitations in UAV-based road extraction. Scene variability, shadows, non-uniform road surfaces, and confusion with similar structures such as rooftops or rivers make them unreliable. In addition, they depend heavily on handcrafted features and fixed thresholds, which limits their adaptability across different datasets [52].

Figure 2.1 shows examples of classical segmentation results on UAV imagery, including global thresholding, Canny edge detection, Otsu thresholding, watershed segmentation, and k-means clustering. As can be seen, these methods often give fragmented, noisy, or over-segmented outputs because of shadows, occlusions, and spectral ambiguities. These problems highlight the weaknesses of traditional approaches and explain why more robust, data-driven deep learning methods are needed. The next section will focus on these methods in detail.

Unlike traditional methods, deep learning approaches-based segmentation techniques have become strong alternatives because they can learn features directly from data. Convolutional Neural Networks (CNNs), especially encoder–decoder designs like U-Net, have shown very good results in tasks that need precise localization, such as road extraction [15, 16]. The U-Net model combines downsampling and upsampling paths with skip connections, which helps it keep spatial details while still capturing context. This makes it effective for high-resolution aerial images.

The performance of these models depends a lot on the training setup, including normalization, batch size, and data diversity [17, 21, 23]. Batch Normalization (BN) is widely used but can be unstable with small batch sizes, which are often required for large UAV images due to memory limits. Group Normalization (GN) reduces some of these problems but does not fully use information across samples. For this reason, hybrid methods such as Batch Group Normalization (BGN) were proposed. Adding BGN to U-Net improves

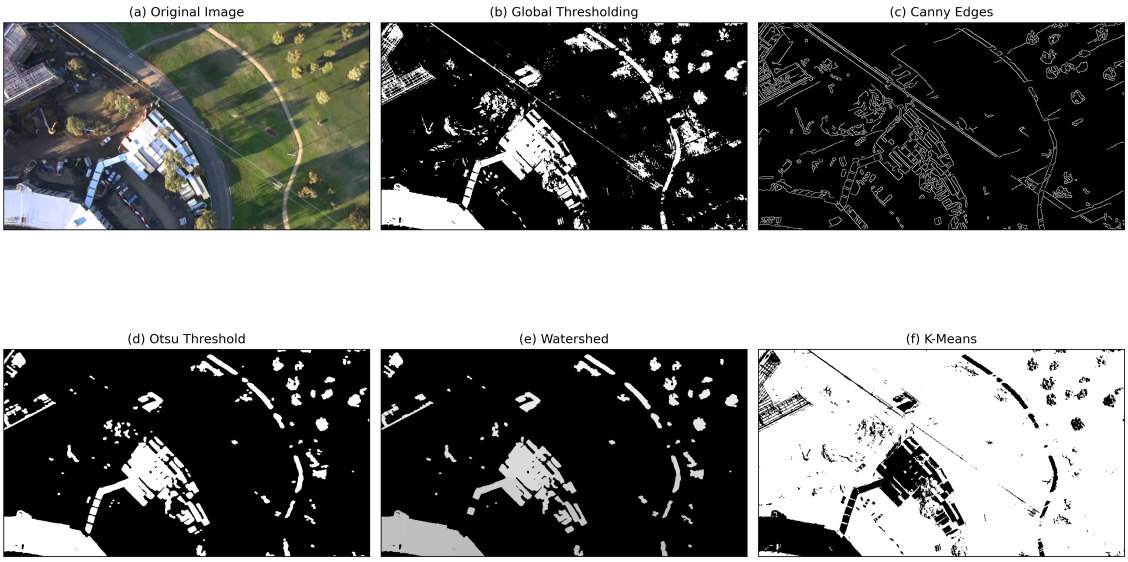


Figure 2.1: Comparison of classical segmentation methods applied to UAV imagery: (a) Original image; (b) Global thresholding; (c) Canny edge detection; (d) Otsu thresholding; (e) Watershed segmentation; (f) K-means clustering. These classical methods often struggle with fragmented edges, noise, and over-segmentation due to shadows and complex backgrounds.

stability during training and increases segmentation accuracy, especially under UAV constraints.

In summary, traditional segmentation methods are important but limited when applied to UAV scenes. Deep learning models, supported by adaptive normalization strategies, provide more reliable and scalable solutions. This shift from classical to methods that learn from data motivates the design of models that achieve high accuracy and robustness in UAV road extraction [53, 45].

2.5 Recent Architectures and Improvements

Over the years, researchers have tried to push the limits of road extraction by designing more advanced and hybrid neural networks. One line of work focused on Generative Adversarial Network (GAN)s, which are able to produce realistic outputs by learning through competition between a generator and a discriminator. Building on this idea, Zhang et al. combined Deep Convolutional Generative Adversarial Network (DCGAN) and Conditional Generative Adversarial Network (CGAN) with a fully convolutional generator, which helped to improve the spatial coherence of road extraction results [54, 55, 56].

Another direction explored the use of recurrent connections. For instance, Yang et al. proposed Region-based Convolutional Neural Network (RCNN)-U-Net, where recurrent links were added to CNN backbones so that the model could capture both spatial patterns and sequential dependencies [57]. VNet approached the problem from another angle by

designing a Cross-Entropy-Dice Loss (CEDL) to reduce class imbalance and better capture narrow road segments [58]. These solutions showed clear improvements, but their heavy computational cost limited their use in real-time UAV applications [45].

More recently, researchers have turned to Transformer-based models. By combining the global attention of Vision Transformers with the local detail extraction of CNNs, hybrid CNN-Transformer networks achieved more accurate segmentation, especially when fine details and large structures had to be recognized together [59]. Yet, these benefits came at the price of longer training times and slower inference speeds [14].

Attention mechanisms have also been integrated into traditional encoder–decoder designs. UNet++, for example, was enhanced with the Convolutional Block Attention Module (CBAM), which helped the network focus on important regions while reducing false positives in cluttered scenes [60, 61]. Similarly, Multi-Scale Attention Generative Adversarial Network (MS-AGAN), a multi-scale adversarial model, improved road continuity and connectivity, though challenges like occlusion and scene clutter remained [55].

Taken together, these advances show how the field is moving toward hybrid solutions that combine the strengths of CNNs, Transformers, and adversarial learning. Wang et al. [62], for instance, use a dual-branch fusion encoder with spatial-channel attention to capture both global context and fine details, while Zhang et al. [55] leverage adversarial training at multiple scales to strengthen road continuity. Hybrid GANs [56] and recurrent CNNs [63] add further improvements in spatial consistency, although they remain computationally demanding. Despite this progress, extracting roads from UAV images, especially in dense urban environments, continues to pose difficulties. This ongoing challenge keeps driving research toward adaptive architectures and refined loss functions [64, 65].

2.6 Representative UAV Road Detection Algorithms (Pre-2015 to 2025)

In recent years, road detection from Unmanned Aerial Vehicle (UAV) imagery has progressed significantly, supported by advances in remote sensing and the growing need for automated infrastructure monitoring and traffic management [53, 55]. Early studies before 2015 mainly relied on classical image processing techniques such as edge detection, thresholding, morphological operations, and Hough transforms [25, 44, 49]. These approaches attempted to identify roads using handcrafted rules based on color, texture, and geometry. However, as we observe from the literature, their performance was limited by shadows, occlusions, and illumination changes. Roads often shared visual similarities with objects such as rooftops or riverbanks, which further reduced their reliability [45, 52]. Consequently, we can see that classical methods lacked robustness and adaptability,

restricting their deployment in large-scale or complex urban environments.

The emergence of deep learning, particularly Convolutional Neural Networks (CNNs), brought a decisive shift around 2015. Architectures like Fully Convolutional Networks (FCNs), SegNet, and U-Net enabled pixel-wise segmentation by learning hierarchical representations directly from data [66, 36, 11]. U-Net, with its encoder–decoder structure and skip connections, proved especially powerful in capturing both spatial detail and broader context, which made it highly effective for aerial imagery [11, 67, 15]. However, from our perspective, these models come with important constraints: they require extensive annotated datasets and high computational resources, both of which can be challenging for UAV platforms with limited onboard capacity [68, 69]. In addition, we note that performance can be sensitive to training hyperparameters such as batch size and normalization, which are critical considerations in UAV applications [17, 21, 23].

To mitigate these issues, researchers have turned toward task-specific and lightweight models. For example, Sunitha et al. designed a framework tailored for UAV-based road surface damage detection, incorporating domain-specific features to improve accuracy [70]. While effective in some cases, it showed limited generalization and reduced suitability for real-time use. Zhang et al. explored MobileNet to cut down computational demands, and although this improved efficiency, it compromised the extraction of fine spatial details in dense scenes [71]. From our analysis, this underlines a recurring trade-off between compact models and segmentation fidelity.

More recently, hybrid and transformer-based models have been proposed. By combining CNNs with attention mechanisms, these approaches aim to capture long-range dependencies and fine-scale details simultaneously. Liu et al., for instance, introduced a hybrid CNN–Transformer model for better detection of narrow roads and markings [72]. Although such designs are promising, we find that their computational and training complexity still restricts deployment on UAVs with limited hardware [71, 59].

UAV imagery is also being applied beyond road segmentation. For example, Cruzado et al. employed background subtraction techniques for freeway incident detection, reporting accuracies up to 92% while also showing how UAV altitude and viewing distance affect detection performance [73]. Multi-object detection models such as HeMoDU have also been developed, demonstrating improved detection speed and accuracy in challenging urban scenes when tested on datasets like VisDrone2019 and UA-DETRAC [74].

In parallel, we observe growing interest in transfer learning and ensemble approaches. Senthilnath et al. proposed Deep TEC, which combines transfer learning with ensemble classifiers. Their work showed how pre-trained models such as conditional GANs, CycleGANs, and FCNs could be adapted to UAV imagery to improve generalization [75]. From our perspective, this line of research highlights the benefit of reusing knowledge from related domains to address dataset variability.

Another emerging trend is the application of modern object detection frameworks such

as YOLOv7 and YOLOv8 for UAV-based road damage detection. These models achieved accurate and real-time classification of different road damage types, providing UAV platforms with reliable tools for infrastructure assessment [76, 77]. Evaluations were typically carried out with precision, recall, and F1-scores, showing their practical efficiency.

More integrated frameworks have also been explored. For example, C-DeepLabV3+ was introduced to perform both road segmentation and vehicle detection by incorporating coordinate attention and cascade feature fusion modules [78]. We consider these multi-task approaches an important step toward broader scene understanding, which is essential for intelligent transportation systems [79].

In summary, the field has evolved from handcrafted methods to increasingly sophisticated deep learning and hybrid solutions [53, 45]. Despite these advances, challenges remain, including dependence on large annotated datasets, UAV hardware constraints, and the ongoing need to balance model complexity with real-time applicability [17, 21]. These limitations motivate our own work, where we aim to design methods that are both effective and practical for UAV-based road extraction.

To address this, we propose the *BGN-U-Net* model, which integrates Batch Group Normalization into the U-Net architecture. With this design, we aim to improve training stability and segmentation accuracy across varying batch sizes. Our approach seeks to provide a reliable and efficient solution for UAV road detection, bridging the gap between research advances and real-world deployment. The following chapter presents our methodology in detail, including dataset preparation, model architecture, and evaluation. Table 2.1 summarizes key UAV road detection algorithms and approaches developed between 2015 and 2025, highlighting their main contributions and application focus.

Table 2.1: Representative UAV Road Detection Algorithms Developed Between 2015 and 2025

Year	Algorithm / Approach	Description & Contribution	Application Focus
Before 2015	Classical Methods	UAV road detection relied on classical image processing techniques such as edge detection, color thresholding, and Hough transform, limited by computational resources and dataset size.	Road segmentation from aerial imagery [80]
2017–2018	CNN-based Segmentation	Emergence of CNN models (e.g., U-Net variants) adapted for UAV imagery, improving accuracy and robustness in complex urban scenes.	Road segmentation [36]
2019–2020	Background Subtraction	Background subtraction applied to UAV video for vehicle detection, achieving up to 92% accuracy and analyzing UAV positioning effects on detection performance.	Vehicle detection and traffic monitoring [73]
2021–2022	YOLO-based Detection	Adoption of YOLO architectures (YOLOv5, YOLOv7) for real-time road damage and vehicle detection, enabling fast and accurate UAV monitoring.	Road damage and vehicle detection [81]
2023–2024	C-DeepLabV3+ + S-YOLOv5	Integrated framework combining coordinate attention enhanced DeepLabV3+ for road segmentation and SimAM attention enhanced YOLOv5 for vehicle detection, achieving state-of-the-art results.	Simultaneous road segmentation and vehicle detection [78, 79]
2025	Transformer-CNN Framework	Incorporation of Transformer modules significantly improving road damage detection accuracy on UAV datasets, reflecting the trend toward transformer-based architectures.	Road damage detection [82]

2.7 Normalization Techniques

When training deep neural networks, we often face difficulties such as vanishing and exploding gradients or internal covariate shift, where the distribution of activations changes during training. These problems can slow convergence and destabilize learning. Over time, researchers have developed several strategies to address them, and among these, normalization techniques have proven especially effective in improving stability and speeding up optimization.

The main idea of normalization is to standardize intermediate feature distributions

inside network layers. By doing this, we reduce internal covariate shift and make optimization more efficient. Since the introduction of Batch Normalization (BN) by Ioffe and Szegedy in 2015 [17], normalization has become a core component of modern architectures. BN works by normalizing activations within each mini-batch using their mean and variance, followed by a scaling and shifting step with learnable parameters. This procedure stabilizes training, accelerates convergence, and often improves generalization performance.

Batch Normalization effectively mitigates internal covariate shift by normalizing layer inputs within each mini-batch during training. As shown in Figure 2.2, BN accelerates training and improves final accuracy on the MNIST dataset. The figure illustrates that networks trained with BN achieve higher test accuracy more rapidly than those without. Furthermore, the evolution of input distributions to a sigmoid activation reveals that BN maintains a more consistent distribution over training iterations, reducing internal covariate shift. This stabilization enables the use of higher learning rates and improves gradient flow, ultimately enhancing model performance.

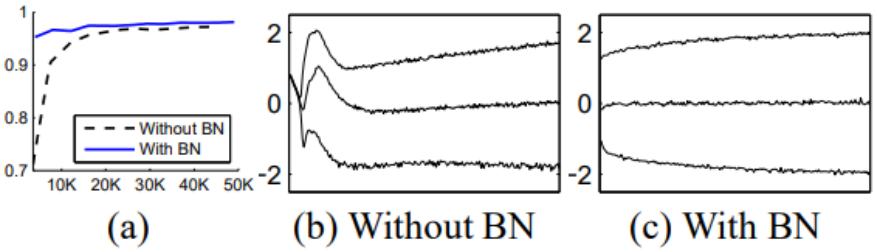


Figure 2.2: (a) Test accuracy of a neural network trained on MNIST with and without Batch Normalization (BN) versus training steps. BN accelerates training and improves accuracy. (b, c) Evolution of input distributions to a typical sigmoid activation during training, shown at the 15th, 50th, and 85th percentiles. BN stabilizes these distributions, reducing internal covariate shift. Figure adapted from Ioffe and Szegedy (2015) [17].

Still, BN is not without limitations. During inference, when the model processes single samples rather than batches, it must rely on running statistics collected during training. If the test distribution differs from the training distribution, performance can suffer. This has motivated alternative approaches that avoid dependency on batch statistics.

Among these alternatives, Layer Normalization (LN) [22] normalizes across all features within a single sample, eliminating dependence on batch size and making it particularly suitable for recurrent architectures. The benefits of LN were demonstrated using the Deep Recurrent Attention Writer (DRAW) model on the MNIST dataset. As shown in Figure 2.3, the model with LN converges almost twice as fast as the baseline model without normalization. Both models eventually reach similar performance levels, but LN accelerates learning by stabilizing internal computations during training, improving efficiency without sacrificing accuracy.

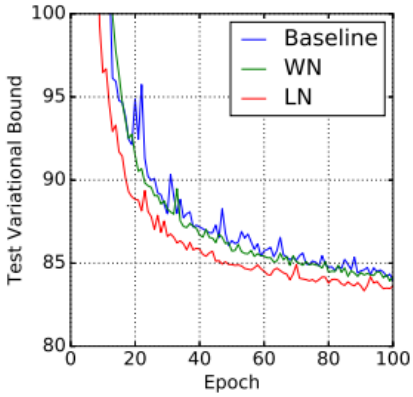


Figure 2.3: DRAW model test negative log likelihood with and without layer normalization [22].

Building upon the concept of per-sample normalization, Instance Normalization (IN) [83] normalizes each channel of individual samples independently. This approach has been extensively applied in style transfer tasks due to its ability to remove instance-specific contrast information. While IN effectively normalizes each channel, it does not explicitly leverage relationships among channels.

To address this, Group Normalization (GN) [21] divides channels into groups and normalizes within each group independently of the batch dimension. This design overcomes batch size limitations and maintains stable training dynamics even with small or varying batch sizes, making GN especially suitable for memory-constrained environments such as UAV image segmentation.

Further developments include Weight Normalization (WN) [84], which reparameterizes neural network weights to decouple magnitude and direction, facilitating faster convergence without relying on batch statistics. Additionally, hybrid methods such as Batch Group Normalization (BGN) [23] combine the advantages of BN and GN by computing normalization statistics across both batch and group dimensions. BGN preserves rich feature representations while maintaining robustness to varying batch sizes, proving particularly effective in high-resolution semantic segmentation tasks.

Other sophisticated normalization frameworks, including Switchable Normalization (SN) [85] and Batch-Instance Normalization (BIN), dynamically learn to balance between BN, IN, and LN statistics. These methods adapt normalization strategies on a per-layer basis to optimize performance across diverse tasks and batch configurations.

As illustrated in Figure 2.4, normalization techniques differ primarily in the dimensions over which they compute statistics, which significantly affects their behavior and performance in deep neural networks.

Incorporating advanced normalization methods such as GN and BGN into segmentation networks like U-Net has been shown to improve training stability and segmentation accuracy, especially under constraints posed by small batch sizes and high-resolution in-

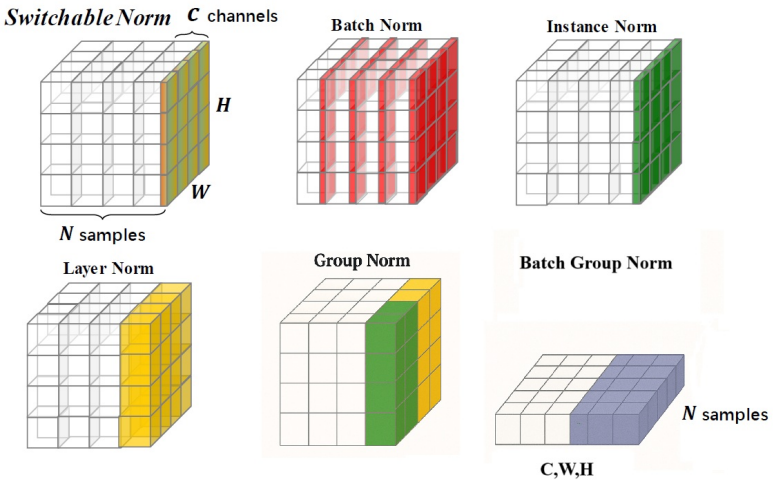


Figure 2.4: Visualization of different normalization techniques in deep learning, including Batch Normalization, Instance Normalization, Layer Normalization, Group Normalization, Batch Group Normalization, and Switchable Normalization. Each colored region illustrates the scope over which statistics are computed for each normalization method.

puts typical of UAV-based semantic segmentation. This thesis further investigates the application of BGN within U-Net to evaluate its effectiveness in this context.

To better understand the distinctions and practical considerations of the various normalization methods discussed, Table 2.2 summarizes their key characteristics. The table compares each technique based on the dimensions over which normalization is performed, dependence on batch statistics, typical application domains, and principal advantages. This overview highlights how different normalization strategies address specific challenges in deep neural network training, such as handling small batch sizes, stabilizing recurrent computations, or improving style transfer. This comparison provides a useful reference for selecting appropriate normalization methods tailored to particular architectures and tasks.

2.8 Research Gaps and Motivation

Despite the progress made in UAV-based image segmentation, important challenges remain before models can be used reliably in real-world situations. Many advanced deep learning models demand significant computational power and memory, which makes them difficult to run on UAV platforms with limited onboard hardware [86, 87]. This creates a trade-off between accuracy and efficiency that is problematic for time-sensitive tasks such as disaster response or autonomous navigation.

The accuracy and robustness of current models can also decrease in difficult conditions. Occlusions, changes in illumination, and cluttered environments often reduce segmentation quality, especially when annotated training data is scarce or imbalanced [88,

Table 2.2: Comparison of Normalization Techniques in Deep Neural Networks

Normalization	**Normalization Dimension**	**Batch Dependence**	**Typical Applications**	**Key Advantages**
Batch Normalization (BN) [17]	Across batch and spatial dimensions per channel	Yes, depends on mini-batch statistics	CNNs, general deep networks with large batch sizes	Stabilizes training, accelerates convergence, improves generalization
Layer Normalization (LN) [22]	Across all features within each individual sample	No, independent of batch size	Recurrent Neural Networks, Transformers	Suitable for variable or small batch sizes, stabilizes recurrent computations
Instance Normalization (IN) [83]	Per channel, per individual sample	No	Style transfer, GANs	Removes instance-specific contrast, preserves style information
Group Normalization (GN) [21]	Groups of channels within each sample	No	Object detection, segmentation, small/variable batch sizes	Stable training with small batches, balances between BN and LN
Weight Normalization (WN) [84]	Reparameterizes weight vectors (magnitude and direction)	No	General deep networks	Faster convergence without batch statistics
Batch Group Normalization (BGN) [23]	Combines batch and group normalization statistics	Yes	High-resolution semantic segmentation	Robust to batch size variation, preserves rich features
Switchable Normalization (SN) [85]	Learns weighted combination of BN, IN, LN	Yes/No (adaptive)	Diverse tasks	Adaptive normalization, optimizes per layer for task

89]. Since creating large, high-quality UAV datasets is costly and time-consuming, this remains a practical limitation.

Normalization techniques are another key element for stable and efficient training. However, widely used methods such as Batch Normalization rely on large batch sizes to perform well [17]. In UAV workflows, batch sizes are often small or variable because of hardware limits or dataset constraints, which reduces the effectiveness of BN [21, 23]. Alternatives such as Group Normalization and Batch Group Normalization have been introduced, but there is still a lack of models that integrate normalization strategies specifically adapted to UAV scenarios.

These challenges point to the need for lightweight segmentation models, defined as architectures that require fewer parameters and computational resources, while still balancing accuracy, stability, and efficiency and remaining suitable for small-batch learning. Batch Group Normalization (BGN) is a promising solution because it combines the strengths of both batch and group statistics [23], making it particularly relevant for UAV applications.

Looking at the broader literature, road extraction has evolved from classical image processing approaches to deep learning-based methods. Among these, U-Net and its many variants have become widely used because of their flexibility and strong performance [11, 38, 90]. Yet, these models do not always address UAV-specific challenges, such as varying environmental conditions or limited batch sizes. While normalization has improved the training stability of these architectures, further work is needed to adapt such methods more directly to UAV-based segmentation [21, 23, 85].

In this thesis, we aim to fill these gaps by developing lightweight segmentation mod-

els that incorporate Batch Group Normalization. Our objective is to improve robustness, efficiency, and generalization in UAV-based road segmentation, while ensuring that the proposed models remain practical for deployment in real-world scenarios.

Chapter 3

Methodology

Contents

3.1 Introductinon **23**

3.2 Memory Requirements to Train a Deep Learning Model **23**

3.3 Dataset **26**

3.4 Data Preprocessing **26**

3.5 Data Augmentation **29**

3.6 Normalization Techniques **30**

3.7 Baseline Model **33**

3.8 Proposed Model Architecture **33**

3.9 Implementation and Training Setup **35**

3.10 Evaluation Metrics **36**

3.11 Summary **36**

3.1 Introductinon

In this chapter, we present the methodology we followed to tackle the task of road segmentation in aerial images captured by Unmanned Aerial Vehicles (UAVs). We begin by describing the dataset we used, highlighting its key characteristics and explaining why it is well-suited to address the problem. We then detail the preprocessing steps we applied to the raw images, including normalization and geometric corrections, to ensure that the data is consistent and of high quality. To further improve our model's ability to generalize, we also explore several data augmentation techniques that artificially increase the diversity of training samples.

Next, we present the normalization strategies we explored, paying particular attention to their effect when training with small batch sizes, which is a common limitation when handling high-resolution UAV imagery. We then describe the architecture of our proposed segmentation model, which builds upon the well-established U-Net framework. Our adjustments to the baseline are specifically designed to improve robustness and accuracy, especially through the integration of a tailored normalization technique that addresses the unique challenges posed by UAV data.

We provide full details of our training setup, including the optimization algorithms and learning rate schedules, to ensure reproducibility and demonstrate the rigor of our experiments. Finally, we describe the evaluation metrics used to quantitatively assess segmentation performance, allowing for a thorough and objective comparison with existing methods. Overall, our goal is to develop a segmentation framework that can accurately and reliably extract road networks from UAV imagery, contributing to advancements in remote sensing and computer vision applications.

3.2 Memory Requirements to Train a Deep Learning Model

When training a deep learning model on large images, memory usage can far exceed the raw size of the input data. The total memory required depends on several factors including: image resolution, color depth, the number of channels, model architecture, sizes of intermediate feature maps, batch size, and data precision. Being able to estimate these memory requirements accurately is essential for managing resources efficiently and avoiding out-of-memory errors during training. Recent studies have introduced optimization strategies, such as tensor lifetime analysis and checkpointing, which help reduce peak memory consumption without affecting model accuracy or training time [91, 92, 93]. Additionally, analyses examining memory use in relation to hyperparameters have shown that batch size and model depth play a major role in determining memory demands [94].

The memory required to store a single input image can be estimated by multiplying its width, height, number of channels, and bytes per channel. While standard images are often

stored as 8-bit per channel (1 byte), deep learning frameworks typically represent pixel values and intermediate activations using 32-bit floating-point precision (float32), which requires 4 bytes per value [95]. Therefore, for practical training memory calculations, the bytes per channel is usually taken as 4 bytes.

The memory M required to store a single input image can be estimated as:

$$M = W \times H \times C \times B \tag{3.1}$$

where W and H are the image width and height in pixels, C is the number of channels, and B is the bytes per channel.

For example, a single 1028×1028 RGB image stored as 8-bit data requires approximately 3.02 MB of memory, while the same image stored as float32 data requires roughly four times more memory, approximately 12.1 MB.

However, the input image memory represents only a small fraction of the total memory consumed during training. The dominant memory usage arises from storing intermediate feature maps (activations) at each layer of the network. Each layer outputs feature maps with specific spatial dimensions and channel counts, and the memory required to store these activations accumulates across all layers. The memory needed to store feature maps for a single image can reach approximately 76 MB for a 256×256 image, 1.2 GB for a 1024×1024 image, and nearly 9.6 GB for a $4K$ image. These values reflect only the forward pass; the backward pass during training will consume a similar amount of memory [95].

Formally, the activation memory $M_{activation}$ for an input image of width W and height H can be estimated by scaling from a baseline memory M_{base} at resolution $W_{base} \times H_{base}$:

$$M_{activation} = M_{base} \times \frac{W \times H}{W_{base} \times H_{base}} \tag{3.2}$$

where M_{base} is the activation memory at the baseline resolution. This formula assumes the network architecture and number of channels remain constant.

It is important to note that the backward pass approximately doubles the memory requirement since gradients of activations must also be stored [96]. Therefore, efficient memory management strategies are essential for training high-resolution models on limited hardware.

In addition to activation memory, the model's trainable parameters contribute to the overall memory footprint. The BGN-UNet architecture used in this thesis contains approximately 31 million trainable parameters, which require additional memory storage. These parameters remain constant regardless of the input image size but add to the total memory demand during training and inference.

During training, activation memory requirements scale linearly with the batch size, as activations for all images in the batch must be stored simultaneously. Additionally,

memory is required for gradients and optimizer states, which typically adds a factor of two to three times the activation memory, depending on the optimizer used [95, 96, 97].

In the context of this thesis, we trained the BGN-UNet model on images from two different sequences, each subjected to distinct preprocessing steps. After applying reflection padding, the images in the first sequence have a size of 1024×512 pixels, while the images in the second sequence retain their original resolutions (1280×720 pixels and 1024×576 pixels), as no padding was applied. The memory needed to store the activations for a single image as it passes through the BGN-UNet model depends directly on these input dimensions. Naturally, larger images require more memory to hold the intermediate feature maps generated at each layer, including the encoder, bridge, and decoder blocks. For comparison, if the full images were processed directly by the network, we estimated the activation memory footprint to be roughly 1.1 GB for the 1024×512 images, and slightly higher for the larger images in the second sequence. As we increase the batch size, the total memory demand grows linearly, which makes careful planning of computational resources essential during training.

To manage these substantial memory demands, several strategies were employed. First, all RGB images were converted to grayscale, reducing the number of channels from three to one and thereby decreasing memory usage by approximately two-thirds [98]. Second, the original high-resolution images were partitioned into smaller patches of 256×256 pixels. This patching reduces the spatial dimensions of feature maps, significantly lowering the per-layer activation memory footprint and enabling the use of larger batch sizes during training [99]. The choice of a patch size of 256×256 has been made for the sake of computational efficiency, striking a balance between capturing adequate spatial information and maintaining a manageable computational load. It is observed in the literature and established practices for similar tasks that a patch size of 256×256 is commonly chosen, reflecting a widely adopted approach in the field [100, 101].

Through these preprocessing steps, we were able to effectively reduce memory usage, which allowed us to train the BGN-UNet model more efficiently and improve overall computational performance. By carefully considering these memory requirements, we ensured that our training process remained practical and scalable, even when working with large, high-resolution images.

In summary, we observed that the memory needed to train a deep learning model on large images depends on several factors, including input size, model architecture, batch size, and data precision. Accurately estimating memory usage across all layers and taking into account the batch size and optimizer overhead proved essential for successful training. In our work, preprocessing strategies such as grayscale conversion and image patching were particularly effective, helping us manage memory demands and enabling the model to learn robustly from the data.

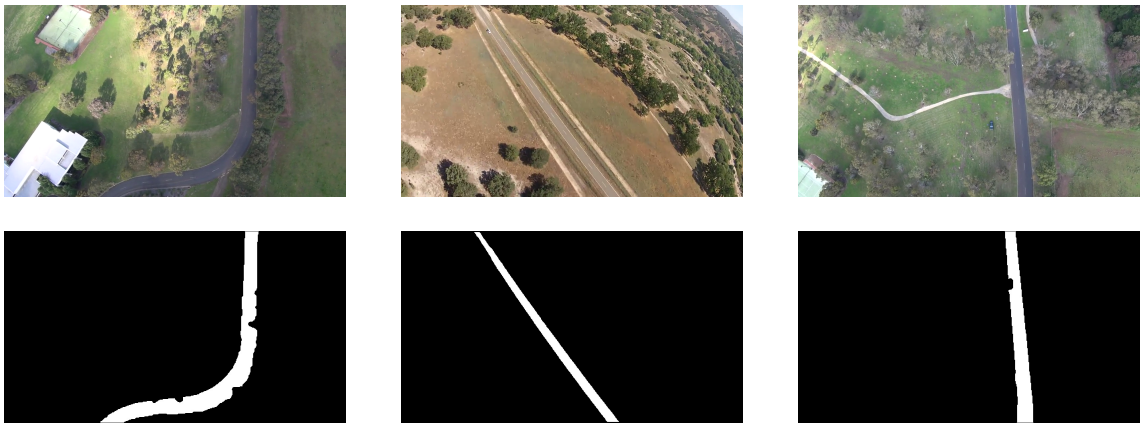


Figure 3.1: Examples of training images and their corresponding ground truth masks.

3.3 Dataset

In our work on road segmentation, we used a dataset consisting of two distinct image sequences. The first sequence includes 224 images, each with dimensions of 848×480 pixels, while the second sequence comprises 109 images, of which 49 have resolution 1280×720 pixels and 60 have resolution 1024×576 pixels. Every image is paired with its corresponding ground truth mask, which labels the road regions to guide the segmentation process. This dataset was originally used in a previous study [102]. We should note that the original source for downloading the dataset (https://sites.google.com/site/hailingzhouwei/) is currently unavailable. In the meantime, we have made the dataset accessible via the following link: `https://drive.google.com/file/d/1DiQBsm5wmN0fFNnZs6i7oG_SHEo6HLej/view?usp=drive_link`.

The dataset includes a variety of road scenarios, which allows us to comprehensively evaluate the model's performance across different types of road environments. Representative examples of training images and their corresponding ground truth masks are shown in Figure 3.1, highlighting the diversity of road layouts and image qualities present in the dataset.

3.4 Data Preprocessing

In our work, we carefully designed a preprocessing pipeline to turn the raw UAV images into a structured and manageable format. This pipeline allowed us to improve the efficiency and performance of our model while also reducing memory usage, as discussed in the previous section [103].

We began by converting the original RGB images to grayscale. We chose this step because it greatly reduced memory requirements by removing the color channels, while still keeping the essential intensity patterns needed for accurate road segmentation. This simplification also made the learning task easier for the model by focusing on the most

important features instead of color information.

Next, the images of the first sequence were processed using reflection padding so that their dimensions became divisible by the selected patch size of 256 pixels. Reflection padding was chosen because it mirrors border pixels, thereby preserving contextual information near image boundaries while reducing artificial edge effects [104]. As a result, all images in the first sequence were resized from 848×480 to 1024×512 pixels, enabling uniform patch extraction.

The second sequence contains images with two native resolutions: 49 images of 1280×720 pixels and 60 images of 1024×576 pixels. Since both resolutions already contain a sufficient number of complete 256×256 regions, no reflection padding was applied. Instead, only complete non-overlapping 256×256 patches were extracted from each image, while incomplete border regions were discarded. This strategy avoids introducing artificial image content while preserving the original image information used for training.

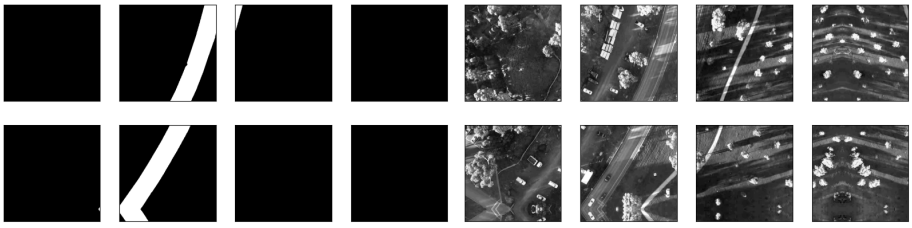
To manage the large image sizes and optimize memory during training, the preprocessed images were divided into non-overlapping patches of 256×256 pixels. This patch-based strategy enabled efficient processing of high-resolution UAV images while increasing the number of training samples. A total of 1792 patches were generated from the first image sequence, whereas 970 patches were extracted from the second sequence. Equation 3.3 and Figure 3.2 illustrate the calculation of the number of extracted patches. We chose the 256×256 patch size because it balanced spatial detail with computational feasibility, as seen in other segmentation studies [100, 101]. Overall, by applying grayscale conversion, sequence-specific image preparation, and patch extraction, we prepared the dataset for efficient training of the proposed model.

$$n = (\frac{SIZE\_X}{patch\_size})(\frac{SIZE\_Y}{patch\_size}) \tag{3.3}$$

$$N_1 = 224 \times \left(\frac{1024}{256}\right) \times \left(\frac{512}{256}\right) = 224 \times 4 \times 2 = 1792 \tag{3.4}$$

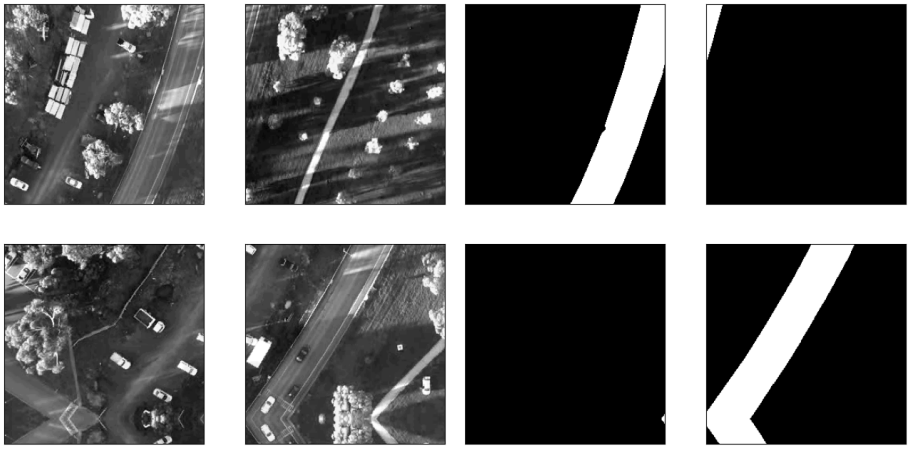
$$\begin{aligned} N_2 &= 49 \times \left\lfloor \frac{1280}{256} \right\rfloor \times \left\lfloor \frac{720}{256} \right\rfloor \\ &\quad + 60 \times \left\lfloor \frac{1024}{256} \right\rfloor \times \left\lfloor \frac{576}{256} \right\rfloor \\ &= 49 \times 5 \times 2 + 60 \times 4 \times 2 \\ &= 490 + 480 \\ &= 970. \end{aligned} \tag{3.5}$$

To improve the quality of our training data, we automatically removed patches that did not contain road information (Figure 3.3). We carried out this pruning by checking the corresponding binary masks, which hold the ground-truth road annotations. Whenever a



(a) (b)

Figure 3.2: Creating Patches. (a): Patches from padded graysclae image, (b): Patches from padded groundtruth



(a) (b)

Figure 3.3: Elimination of Non-Informative Patches. (a):Informative images patches, (b): Informative groundtruth patches

patch had zero road pixels, we excluded it from the dataset. By doing this, we focused the model's learning on the areas that actually matter for road segmentation, reduced noise, and improved overall accuracy.

After extracting and cleaning the patches, we merged the datasets from both sequences into a single dataset of 906 images. We made this decision to increase the variety of our training data by including more types of road structures, environmental conditions, and scene layouts. This diversity helped the model generalize better to different real-world situations.

We then normalized all images to scale the pixel intensity values to a standard range. We included this step because it is required by most neural network architectures and it makes the training process more stable and efficient [105].

By combining patch pruning, dataset fusion, and normalization, we built a dataset that is both memory-efficient and rich in useful information. Because our final dataset was relatively modest in size, we decided to train our U-Net model from scratch. This allowed the model to adapt more closely to our data and reduced the risk of overfitting.

Figure 3.4 shows our complete preprocessing pipeline, starting from the original RGB

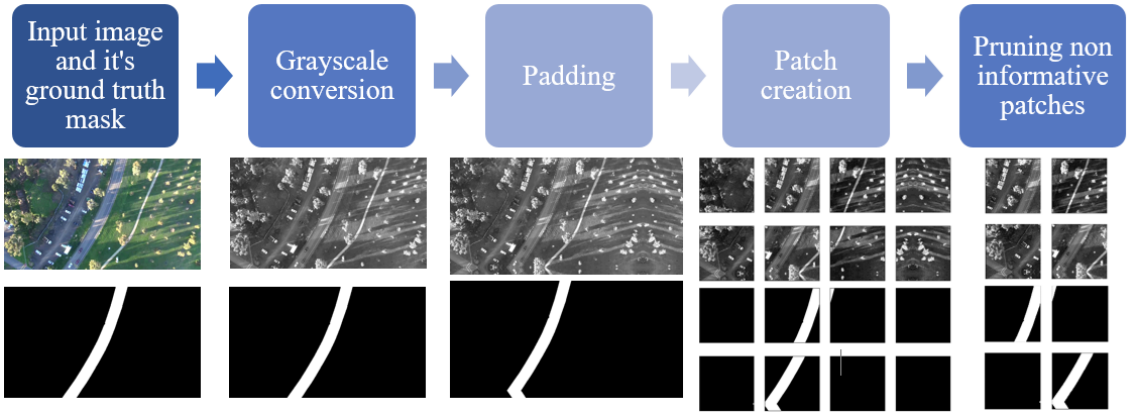


Figure 3.4: Visualization of the preprocessing pipeline applied to the dataset images. The process begins with the input RGB image and its ground truth mask, followed by grayscale conversion, padding, patch creation, and pruning of non-informative patches. Each step is illustrated for both the image and its corresponding mask, providing a clear overview of the transformations performed prior to model training.

images with their corresponding masks, moving through grayscale conversion, padding, patch extraction, and ending with the selection of informative patches we used for training.

3.5 Data Augmentation

Data augmentation played an important role in our work because it allowed us to expand and diversify the training dataset without collecting new images. In deep learning, large and varied datasets are essential for building models that generalize well [106]. In our case, the original dataset contained 906 grayscale images of 256×256 pixels. By applying augmentation techniques, we increased its diversity to better support our U-Net model enhanced with Batch Group Normalization for aerial road segmentation.

Because our dataset was relatively small, we relied on transformations such as rotation and flipping to create additional training samples. These operations added variety without the need for extra labeling, helping us to reduce overfitting, improve robustness, and strengthen the model's ability to generalize [107].

We also applied reflection padding to the first image sequence as part of our preprocessing. This method mirrors the pixels at the edges of the images, extending the borders and providing more spatial context without adding artificial edges. It preserves important edge details and reduces boundary artifacts during convolutions. While some studies warn about possible artifacts when using reflection padding [108, 109], it was applied only to the first sequence, where it was required to make the image dimensions divisible by the selected patch size. For the second sequence, complete 256×256 patches could be extracted directly from the original images; therefore, reflection padding was not applied to

avoid introducing unnecessary artificial border information.

Although our model was trained on a small dataset, combining it with data augmentation allowed us to achieve strong performance. We believe this approach can also scale to larger datasets, which we plan to investigate in future research.

As highlighted in [110], the dataset size required for training depends on the complexity of the task and the number of model parameters. This suggests that smaller datasets can still be sufficient for less complex tasks or models with fewer parameters, supporting our methodological choices.

During training, we applied horizontal and vertical flipping as well as random rotations. We carefully kept the masks aligned with the images so the model could learn from consistent pairs. Figure 3.5 shows one original aerial image and some of its augmented versions. These steps gave us a much more diverse training set, which improved the model's robustness and performance.

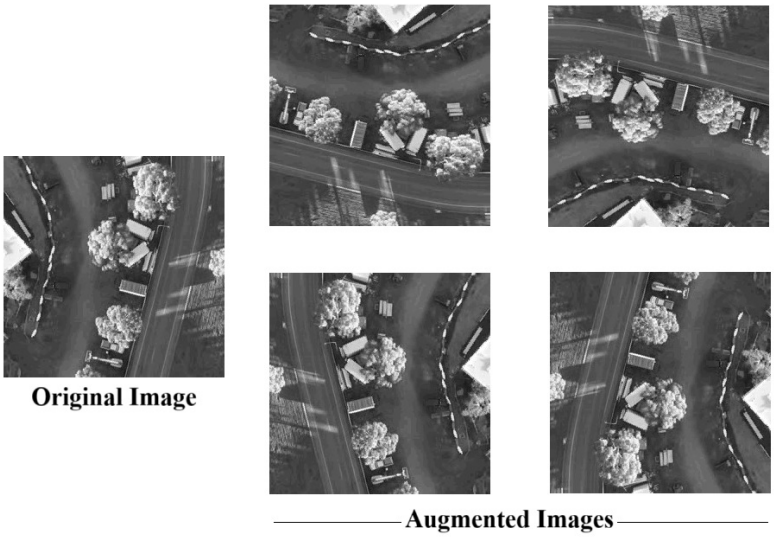


Figure 3.5: Example of data augmentation applied to an aerial image. The original image (bottom left) is shown alongside several augmented versions generated through flipping and rotation. These transformations increase the diversity of the training data and improve model robustness.

3.6 Normalization Techniques

In our work, we gave special attention to normalization techniques because they play a key role in training deep learning models. They help us stabilize the learning process and speed up convergence by keeping the feature distributions consistent across the network. This consistency reduces problems such as internal covariate shift and improves the model's ability to generalize. We focused on three main approaches: Batch Normalization (BN) [17], Group Normalization (GN) [21], and their hybrid, Batch Group Normalization

(BGN) [23]. These techniques are widely used in computer vision and have shown strong results in image segmentation tasks [111].

Batch Normalization operates by standardizing the activations of intermediate layers using the mean and variance computed over the current mini-batch. Formally, given a feature activation x_i, BN computes the normalized activation $\hat{x}_i$ as:

$$x_i' = \frac{1}{\sigma_i}(x_i - \mu_i). \tag{3.6}$$

Here, x represents the feature computed by a layer, and i is an index. In the context of 2D images, $i = (i_N, i_C, i_H, i_W)$ is a 4D vector indexing the features in the order (N, C, H, W), where:

- N is the batch axis,
- C is the channel axis,
- H is the spatial height axis, and
- W is the spatial width axis.

In Batch Normalization, the transformation applied to the input feature to compute the normalized output is given by the following formula:

$$S_i = \{k \mid k_C = i_C\}. \tag{3.7}$$

This means that the mean and variance are calculated along the batch dimension. Thus, normalization is performed across the batch for the same input and makes the optimization process more stable and faster.

While Batch Normalization is very effective in many applications, we found that it performs less well for high-resolution images or small batch sizes, common conditions in segmentation tasks, mainly because of memory limits. To address this, researchers developed Group Normalization (GN), which normalizes features within groups of channels rather than across the whole batch [21]. GN reduces the dependency on batch size and allows us to train more efficiently when memory is limited. The formula for Group Normalization is as follows:

$$S_i = \left\{k \mid k_N = i_N, \left\lfloor \frac{k_C}{C/G} \right\rfloor = \left\lfloor \frac{i_C}{C/G} \right\rfloor \right\} \tag{3.8}$$

where:

- G is the number of groups (pre-defined hyper-parameter, with $G = 32$ by default),
- C/G is the number of channels per group,

- $\lfloor \cdot \rfloor$ denotes the floor operation,
- $\left\lfloor \frac{k_C}{C/G} \right\rfloor = \left\lfloor \frac{i_C}{C/G} \right\rfloor$ means that the indexes i and k are in the same group of channels, assuming each group of channels is stored in a sequential order along the C axis.

Despite the advantages of GN, it does not leverage batch-level statistics, which can sometimes provide useful regularization and improve generalization. To harness the benefits of both BN and GN, Batch Group Normalization (BGN) has been proposed as a hybrid technique that combines the strengths of these methods while mitigating their weaknesses. In Batch Group Normalization (BGN) technique, the channel, height, and width dimensions are initially concatenated into a new dimension, resulting in a flattened representation denoted as $F_{N \times D}$, where $D = C \times H \times W$. The mean μ_g in (4) and variance σ_g^2 in (5) are then computed along both the batch and the new dimension [23]:

$$\mu_g = \frac{1}{NS} \sum_{n=1}^{N} \sum_{d=(g-1)S+1}^{gS} f_{n,d}, \tag{4}$$

$$\sigma_g^2 = \frac{1}{NS} \sum_{n=1}^{N} \sum_{d=(g-1)S+1}^{gS} (f_{n,d} - \mu_g)^2, \tag{5}$$

where $g = 1, \ldots, G$ is a group index used in the group normalization technique and G is the number of groups that the new dimension is divided into, and is a hyper-parameter; $f_{n,d}$ is a member of $F(N)_{N \times D}$, representing a feature instance after merging the channel, height, and width dimensions into a new dimension; $D = C \times H \times W$, where C, H, and W are the channel, height, and width dimensions, respectively. Further, $S = M/G$ is the number of instances indexed inside each divided feature group. The notation gS represents the range of feature instances included within group g for the calculation of the mean μ_g and variance σ_g^2. Specifically, the summation over d goes from $(g-1)S + 1$ to gS, so that each group contains S feature instances along the new dimension D.

BGN dynamically adjusts the number of feature instances used for statistical calculation, employing the group technique from Group Normalization (GN). When dealing with a small batch size, a smaller value for G is chosen to combine the entire new dimension, preventing noisy statistics. Conversely, with a larger batch size, a larger G is selected to partition the new dimension into smaller segments, enabling the calculation of more accurate and less confused statistics. That is why, in our training process, $G = 2$ was used for a batch size of 2 to combine the entire new dimension, and $G = 32$ was used for batch sizes 8 and 16.

In our model, we went a step further and implemented Batch Group Normalization (BGN) as a custom Keras layer. By doing this, we allowed the model to benefit from stable normalization statistics across a wide range of batch sizes. This approach improved training stability, helped the model generalize better, and worked well with the memory

constraints of high-resolution image segmentation [24].

In summary, normalization techniques such as BN, GN, and BGN play a pivotal role in deep learning by stabilizing feature distributions and facilitating efficient training. BGN, in particular, offers a flexible and robust solution by leveraging group-based statistics across flattened feature dimensions, making it especially suitable for scenarios with variable or small batch sizes.

3.7 Baseline Model

We built our baseline model using the standard U-Net architecture, which is widely used for semantic segmentation tasks [11]. This architecture uses a symmetric encoder–decoder design with skip connections to combine low-level spatial information with high-level semantic features. Figure 3.6 shows the original U-Net structure we used as a reference.

In the encoder path, we implemented four levels, each with two 3×3 convolutional layers followed by Rectified Linear Unit (ReLU) activations. We applied 2×2 max-pooling layers with stride 2 to downsample the feature maps, doubling the number of filters at each step (64, 128, 256, 512). At the bottleneck, we added two convolutional layers with 1024 filters.

In the decoder path, we upsampled the feature maps using transposed convolutions with a 2×2 kernel. We then concatenated them with the corresponding cropped feature maps from the encoder through skip connections. Each decoder block contained two 3×3 convolutional layers followed by ReLU activations. Finally, we used a 1×1 convolution with a sigmoid activation to produce the binary segmentation mask.

We implemented this model in TensorFlow/Keras and trained it on grayscale images of $256 \times 256 \times 1$. The total parameter count reached about 31 million, most of which were trainable.

3.8 Proposed Model Architecture

We extended the standard U-Net to build our proposed model, which we call BGN-UNet. This design kept the encoder–decoder structure and skip connections to preserve spatial detail while extracting semantic features at multiple scales [11].

To improve segmentation accuracy and stabilize training, we inserted a Batch Group Normalization (BGN) layer after each convolution. Because the original U-Net does not include normalization, we added BGN to combine the benefits of Batch Normalization and Group Normalization, especially under small batch size conditions.

Our BGN-UNet kept the symmetrical encoder–decoder layout but replaced each standard convolutional block with two 3×3 convolutions followed by BGN (group size 32)

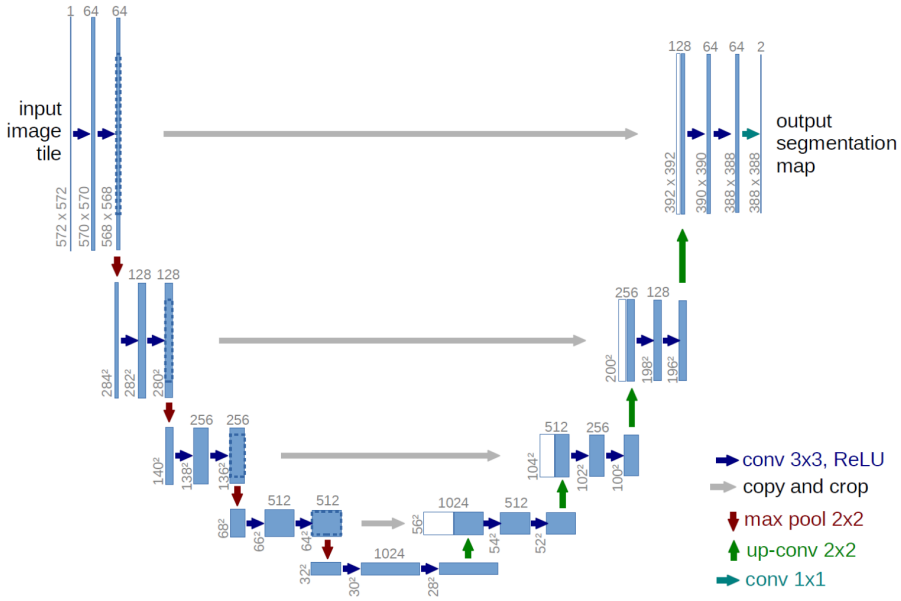


Figure 3.6: The original U-Net architecture illustrating the encoder-decoder structure with skip connections, which forms the baseline model used in this study [11].

and ReLU activations. We still used max pooling for downsampling and transposed convolutions for upsampling. We also kept the skip connections between encoder and decoder layers.

The last layer remained a 1×1 convolution with sigmoid activation to generate the binary segmentation output. We implemented and trained BGN-UNet with the same settings as the baseline model to ensure a fair comparison.

For training, we used 1792 aerial images divided into 256×256 grayscale patches. After removing non-informative patches, we worked with 906 informative patches split into training (80%, 724), validation (10%, 91) and test (10%, 91) sets using the `train_test_split` function. This ensured reproducibility and consistency across all experiments.

Performance improvements achieved with this architecture are presented and discussed in the results section.

Figure 3.7 presents a side-by-side comparison of the convolutional blocks used in the standard U-Net and the proposed BGN-UNet, emphasizing the structural differences and the role of Batch Group Normalization in enhancing the model. Subsequently, Figure 3.8 provides a detailed visualization of the full BGN-UNet architecture, illustrating how these modified blocks are integrated throughout the network. This progressive presentation-from block-level details to the entire architecture-facilitates a clearer understanding of the proposed modifications relative to the original U-Net model [11].

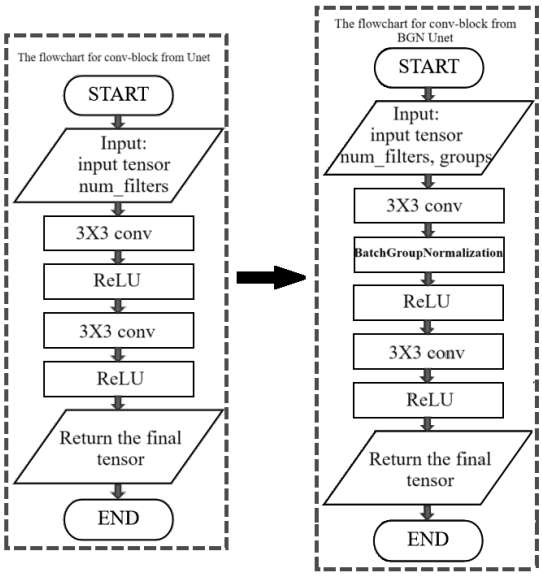


Figure 3.7: Comparison of convolutional blocks in the standard U-Net and the proposed BGN-UNet, highlighting the incorporation of Batch Group Normalization layers in the latter.

3.9 Implementation and Training Setup

The proposed model was implemented using TensorFlow 2.12 within a GPU-enabled Google Colab environment. The experiments were executed on an NVIDIA Tesla T4 GPU with 16 GB of GPU memory using Python 3. Libraries such as NumPy, OpenCV, and Scikit-learn were used to support data preprocessing, visualization, and evaluation metrics. The dataset, which consisted of 906 grayscale UAV images and their corresponding binary masks, was divided into training (80%), validation (10%), and test (10%) sets using a stratified random sampling approach to maintain the overall class balance.

Training was carried out using the Adam optimizer, which combines the advantages of AdaGrad and RMSProp by computing adaptive learning rates for each parameter. This optimizer was chosen for its efficiency and ability to converge faster compared to traditional methods like Stochastic Gradient Descent (SGD). A learning rate of 1×10^{-3} was set for training, which was found to provide a good balance between convergence speed and model accuracy. Binary cross-entropy was used as the loss function, which is suitable for binary semantic segmentation tasks. The model was trained for 25 epochs with a batch size of 16, and a group size of 32 for Batch Group Normalization (BGN) was applied to ensure stable and effective learning throughout the training process.

Our model's total parameter count was 31,065,921, with 31,054,145 trainable parameters and 11,776 non-trainable parameters, resulting in a memory requirement of approximately 118.51 MB. This configuration was designed to maximize performance while stay-

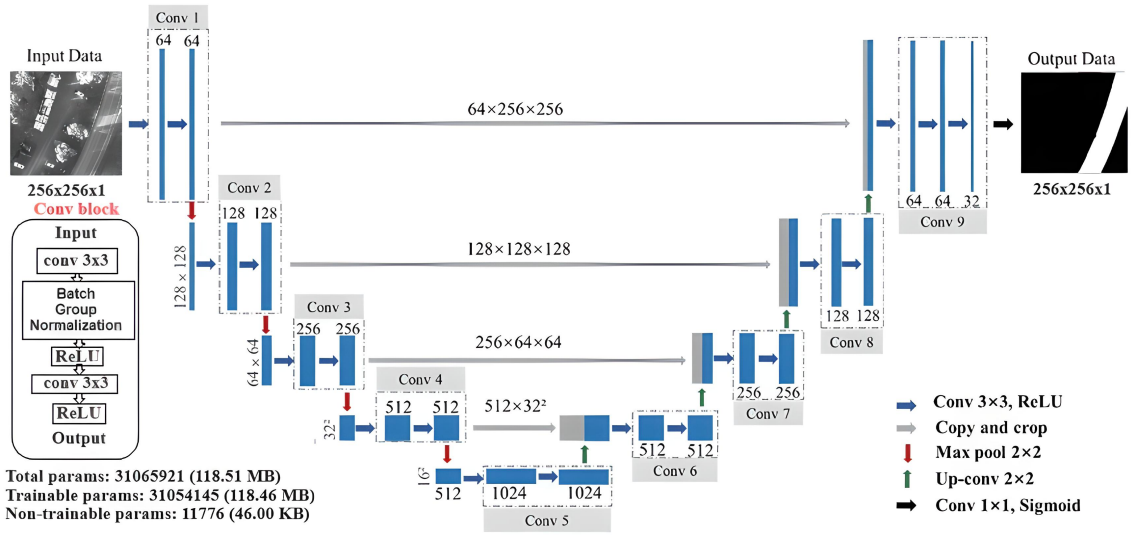


Figure 3.8: Architecture of the proposed BGN-UNet model, based on the original U-Net, with Batch Group Normalization integrated to improve segmentation performance and training stability.

ing within hardware limitations. Training, validation, and test subsets were divided, with 80% used for training, 10% for validation, and 10% for testing, ensuring robust evaluation across all stages.

The hyperparameters were chosen based on preliminary experiments to balance performance and computational efficiency. They were set as shown in Table 3.1 during training.

3.10 Evaluation Metrics

We mainly evaluated our model using the Mean Intersection over Union (Mean IoU), which evaluates the overlap between predicted segmentation and the ground truth. Additional metrics such as accuracy and loss curves were monitored throughout the training process to evaluate convergence behavior and detect signs of overfitting [112].

$$\text{IoU} = \frac{\text{true\_positives}}{\text{true\_positives} + \text{false\_positives} + \text{false\_negatives}} \tag{3.9}$$

3.11 Summary

In this chapter, we presented our full methodology for road segmentation in UAV images. We started by selecting and preparing the dataset, then applied a preprocessing pipeline that included grayscale conversion, reflection padding for the first image sequence, patch extraction, normalization, and removal of non-informative patches. These steps reduced memory use and prepared the data for deep learning.

We then used data augmentation (flipping and rotation) to increase the diversity of our

Parameter	Value
Learning Rate	1×10^{-3}
Loss Function	Binary Crossentropy
Epochs	25
Batch Size	16
Group	32
Optimizer	Adam Optimizer
Validation Split	0.20, Random State = 42
Total Params	31,065,921 (118.51 MB)
Trainable Params	31,054,145 (118.46 MB)
Non-Trainable Params	11,776 (46.00 KB)
Image Data Shape	(906, 256, 256, 1)
Mask Data Shape	(906, 256, 256, 1)
Max Pixel Value in Image	255
Labels in the Mask	[0, 255]
Patch Size	256 x 256 x 1

Table 3.1: Model Hyperparameters

dataset and improve generalization. We analyzed memory requirements and showed how strategies like grayscale conversion and patch-based processing helped us train deeper models on high-resolution images.

We also reviewed normalization techniques and highlighted the limitations of Batch Normalization for small batch sizes. To overcome these challenges, we integrated Batch Group Normalization into the U-Net framework, creating our BGN-UNet model. This model combined the strengths of the original U-Net with advanced normalization to stabilize training and improve segmentation accuracy.

Altogether, this chapter described how we built a solid foundation for evaluating our proposed approach. In the next chapter, we present the results and analysis, showing how well BGN-UNet performs for precise road segmentation in UAV imagery.

The following flowchart (Figure 3.9) summarizes the overall pipeline of the BGN-UNet approach:

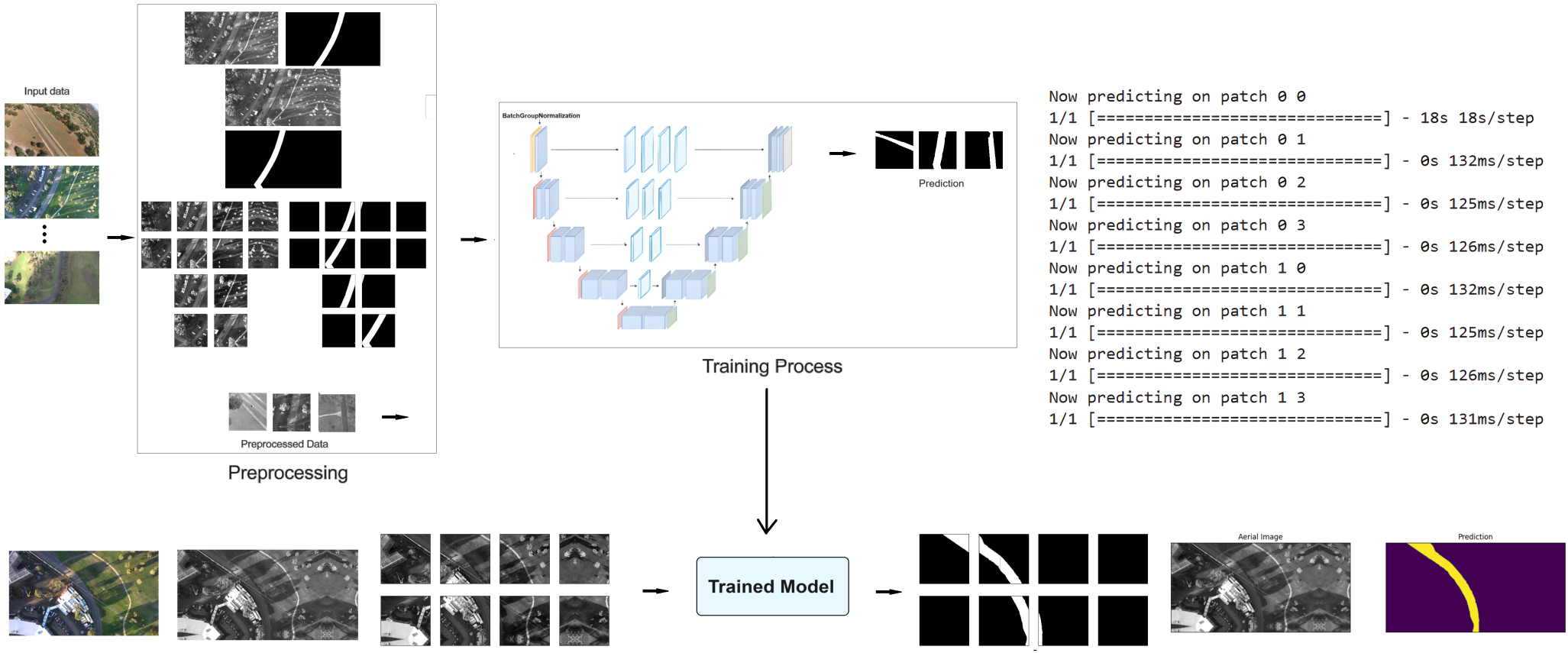


Figure 3.9: Flowchart of the Proposed Methodology of the BGN-UNet approach for UAV road detection.

Chapter 4

Experimental Results and Discussion

Contents

4.1 Introduction **40**

4.2 Experimental Setup **40**

4.3 Quantitative Results **40**

4.3.1 Mean Intersection over Union (IoU) Analysis 40

4.3.2 Inference Speed and Model Initialization 41

4.3.3 Visual Evaluation 42

4.4 Ablation Study: Impact of Batch Size **43**

4.5 Practical Applications of BGN-UNet in Road Detection **48**

4.6 Summary **49**

4.1 Introduction

In this chapter, we present a comprehensive experimental evaluation of various normalization techniques that we implemented within the U-Net architecture for semantic segmentation tasks. We focus on two widely adopted normalization methods—Batch Normalization (BN) and Group Normalization (GN)—which we selected as our baseline approaches because of their proven effectiveness in deep learning models. We also rigorously benchmark our novel Batch-Group Normalization (BGN) technique, which we designed to address the limitations of existing methods under small batch size conditions.

We systematically compare the performance of our BGN-UNet model with BN-UNet and GN-UNet, giving particular attention to segmentation accuracy, convergence behavior, and robustness. We carry out these evaluations in the context of aerial imagery, where high spatial resolution, limited batch sizes, and complex scene variability impose strict requirements on normalization strategies. We rely on both quantitative metrics and qualitative analyses to critically examine the advantages and potential trade-offs of integrating BGN into the U-Net framework.

Through this experimental investigation, we provide insights into the practical applicability and effectiveness of our proposed normalization technique, contributing to the advancement of semantic segmentation methods specifically tailored for UAV-captured aerial data.

4.2 Experimental Setup

We employed the U-Net architecture as the foundation of our work and built three variants: BN-UNet, GN-UNet, and BGN-UNet. We trained each model for 25 epochs using a limited dataset on a single GPU to reflect realistic constraints in computational resources. We used the Adam optimizer because of its proven efficiency and widespread adoption in deep learning applications, which ensured robust convergence and adaptive learning rates [113].

For the binary segmentation task distinguishing between road and non-road classes, we adopted the binary cross-entropy loss function. This choice allowed us to rigorously evaluate the similarity between our model's predictions and the ground-truth labels [114].

4.3 Quantitative Results

4.3.1 Mean Intersection over Union (IoU) Analysis

Performance was primarily assessed using the Mean Intersection over Union (IoU), a standard metric for segmentation accuracy. Experiments were conducted with batch sizes of

Table 4.1: Mean IoU for Different Batch Sizes and Models

Model	**Batch Size 2**	**Batch Size 8**	**Batch Size 16**
BN-UNet	0.96733713	0.96600366	0.97286165
GN-UNet	0.9687331	0.97235453	0.9730376
BGN-UNet	**0.9726844**	**0.9729512**	**0.9740099**
BGN-UNet+data augmentation	**0.98135**	**0.9821**	**0.9840**

2, 8, and 16 to investigate the influence of batch size on each normalization method. The results are summarized in Table 4.1. The IoU for each class is defined as the ratio of the intersection to the union of the predicted and ground truth regions:

$$\text{IoU} = \frac{|\text{Prediction} \cap \text{Ground Truth}|}{|\text{Prediction} \cup \text{Ground Truth}|} = \frac{\text{TP}}{\text{TP} + \text{FP} + \text{FN}} \tag{4.1}$$

The Mean IoU is the average IoU across all classes:

$$\text{Mean IoU} = \frac{1}{K} \sum_{k=1}^{K} \frac{TP_k}{TP_k + FP_k + FN_k} \tag{4.2}$$

[115, 116]

Across all batch sizes, BGN-UNet consistently achieved the highest Mean IoU, indicating superior segmentation accuracy compared to BN-UNet and GN-UNet. These findings underscore the robustness and adaptability of the BGN approach, particularly under varying batch size conditions.

4.3.2 Inference Speed and Model Initialization

Inference speed was evaluated by measuring the time required to predict patches of size 256×256 pixels extracted from a full image of size 1024×512 pixels, which is divided into 8 non-overlapping patches (4 patches horizontally and 2 vertically). It was observed that the initial prediction of the first patch using BGN-UNet required approximately 18 seconds, compared to 12 seconds for BN-UNet and 15 seconds for GN-UNet. This initial delay is primarily due to model initialization overhead, including weight loading, memory allocation, and preprocessing. As shown in Figure 4.1, only the very first patch incurs this significant delay. Subsequent patches are processed almost instantaneously, with inference times dropping to approximately 125–132 milliseconds per patch. This rapid processing after initialization demonstrates efficient reuse of the loaded model and highlights the practicality of BGN-UNet for large-scale inference tasks on high-resolution images.

BN-UNet exhibited the fastest initial processing time, likely due to its reliance on batch statistics, while GN-UNet showed intermediate performance. Despite a slightly longer initialization period, BGN-UNet maintained highly efficient inference for all subsequent

patches.

The initial prediction times for the three models are summarized in Table 4.2.

Table 4.2: Initial Inference Time per Patch for Different Models

Model	Initial Prediction Time (seconds)
BN-UNet	12
GN-UNet	15
BGN-UNet	18

```
Now predicting on patch 0 0
1/1 [==============================] - 18s 18s/step
Now predicting on patch 0 1
1/1 [==============================] - 0s 132ms/step
Now predicting on patch 0 2
1/1 [==============================] - 0s 125ms/step
Now predicting on patch 0 3
1/1 [==============================] - 0s 126ms/step
Now predicting on patch 1 0
1/1 [==============================] - 0s 132ms/step
Now predicting on patch 1 1
1/1 [==============================] - 0s 125ms/step
Now predicting on patch 1 2
1/1 [==============================] - 0s 126ms/step
Now predicting on patch 1 3
1/1 [==============================] - 0s 131ms/step
```

Figure 4.1: Inference times per patch for BGN-UNet on a 1024×512 image divided into 8 patches of 256×256 pixels. The first patch requires approximately 18 seconds due to initialization overhead, while subsequent patches are processed in about 125–132 milliseconds each.

4.3.3 Visual Evaluation

Figures 4.2, 4.3, and 4.4 provide visualizations of training and validation metrics across the three models and various batch sizes. The graphical results for BGN-UNet reveal minimal distortion and negligible false detections, further supporting its effectiveness in segmentation tasks. In Figure 4.5, four randomly selected patches from the test set are showcased, segmented using the BGN-UNet model. These patches present varying levels of difficulty while consistently demonstrating remarkable segmentation results. After training on small grayscale patches of size 256×256, the trained BGN-UNet model is applied to segment larger test images. For inference, a large test image of 1024×576 pixels is first converted to grayscale and then partitioned into non-overlapping patches of the same size used during training. These patches are sequentially fed into the trained model to obtain individual patch predictions. Once all patches have been processed, the predicted

segmentation masks are recombined through an unpatchifying process to reconstruct the full segmentation mask corresponding to the original large image. This patch-based inference strategy enables efficient segmentation of large images while leveraging the model's training on smaller patches, thereby preserving accuracy and computational feasibility as shown in Figure 4.6.

4.4 Ablation Study: Impact of Batch Size

To further elucidate the influence of batch size, an ablation study was conducted, focusing on the BGN-UNet model. The following Mean IoU values were obtained:

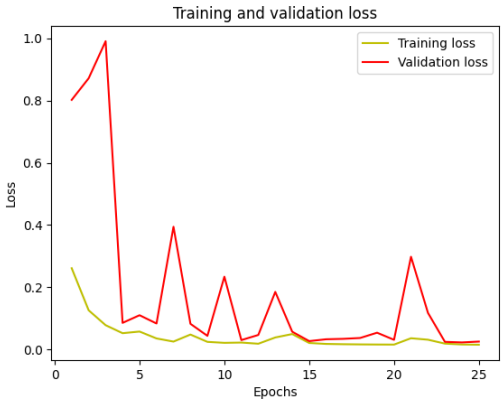
- **Batch Size 2:** 0.98135
- **Batch Size 8:** 0.9821
- **Batch Size 16:** 0.9840

A slight improvement in Mean IoU was observed as batch size increased, with batch size 16 yielding the highest performance. Notably, no evidence of overfitting was detected, and model stability was maintained across all configurations. This suggests that larger batch sizes may contribute to gradient stabilization during training, thereby enhancing performance. However, even with smaller batches, BGN-UNet demonstrated robust results, in contrast to traditional normalization methods, which exhibited greater sensitivity to batch size variations.

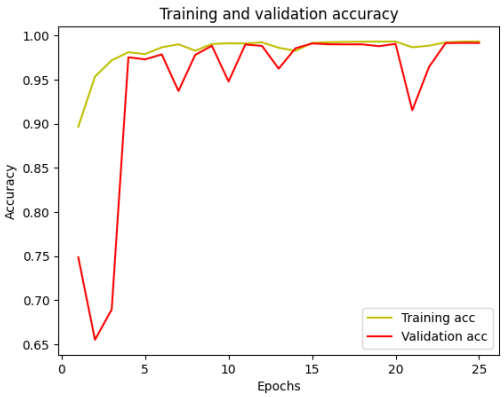
For reference, the performance of BN-UNet and GN-UNet under varying batch sizes is summarized below:

- **BN-UNet:**
  - Batch Size 2: 0.9673
  - Batch Size 8: 0.9660
  - Batch Size 16: 0.9729
- **GN-UNet:**
  - Batch Size 2: 0.9687
  - Batch Size 8: 0.9724
  - Batch Size 16: 0.9730

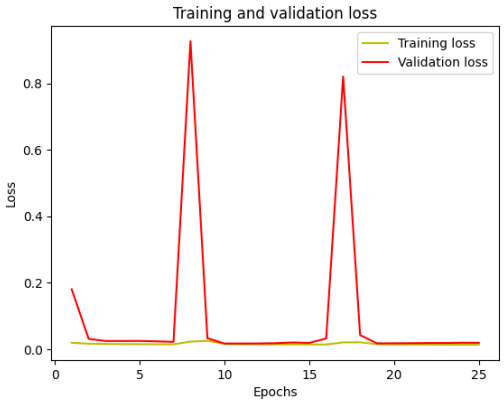
These results reinforce the conclusion that BGN-UNet is less affected by batch size, making it particularly advantageous in scenarios with limited data or computational resources. Future investigations will consider even larger batch sizes (e.g., 32 and 64) to further explore their impact on model dynamics and generalization.



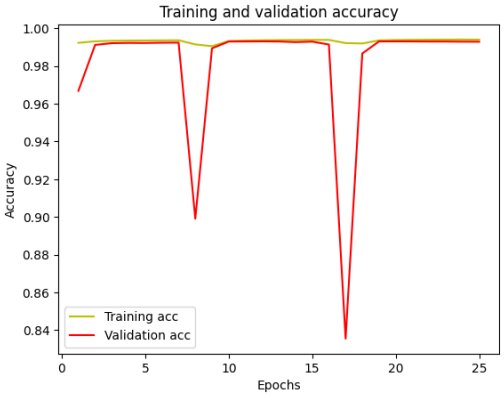
(a) BN-UNet Loss



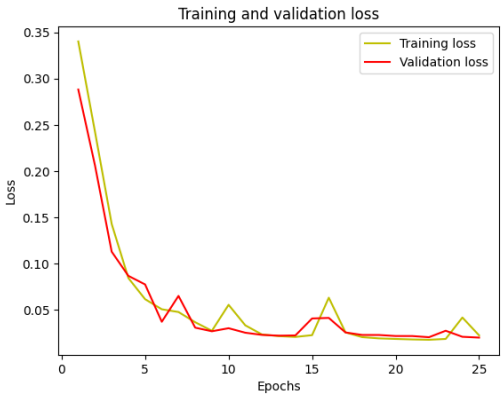
(b) BN-UNet Accuracy



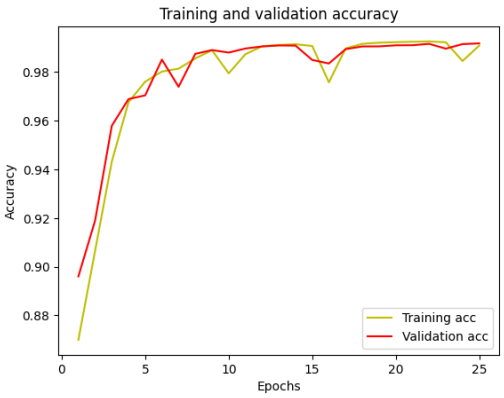
(c) GN-UNet Loss



(d) GN-UNet Accuracy

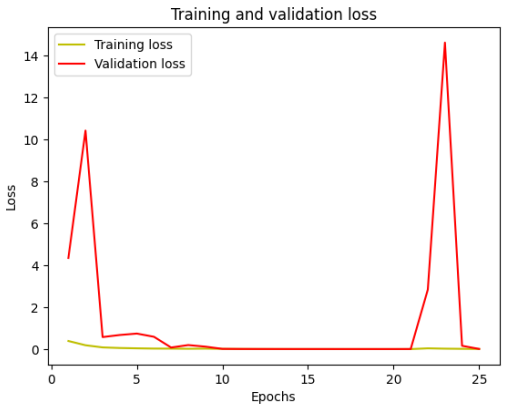


(e) BGN-UNet Loss



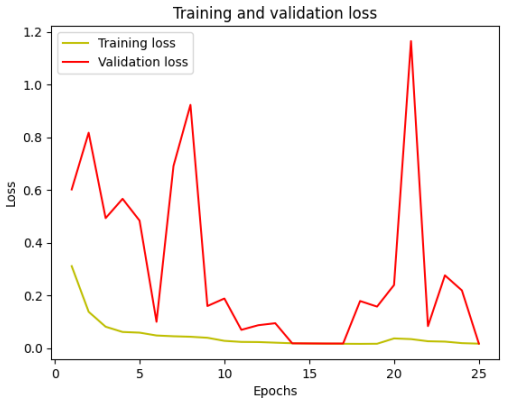
(f) BGN-UNet Accuracy

Figure 4.2: Training and Validation Metrics for Batch Size 2: Loss and Accuracy for BN-UNet, GN-UNet, and BGN-UNet.

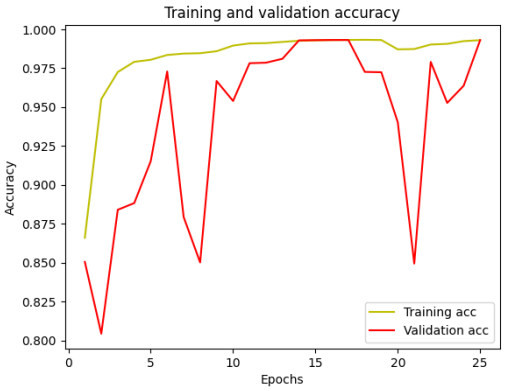


(a) BN-UNet Loss

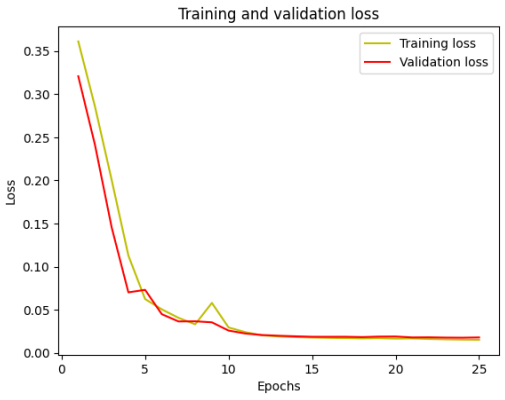
(b) BN-UNet Accuracy



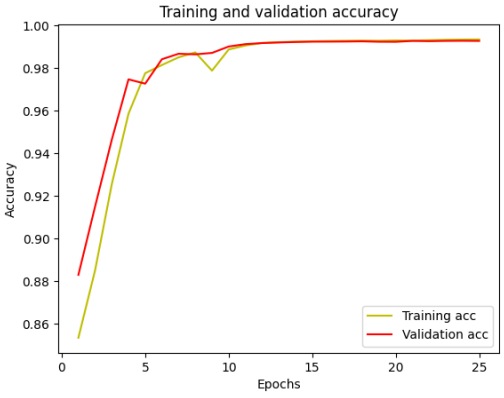
(c) GN-UNet Loss



(d) GN-UNet Accuracy

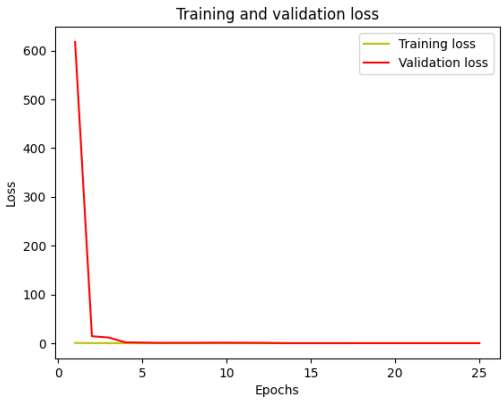


(e) BGN-UNet Loss

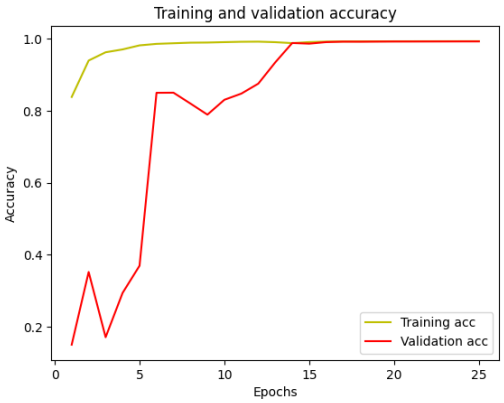


(f) BGN-UNet Accuracy

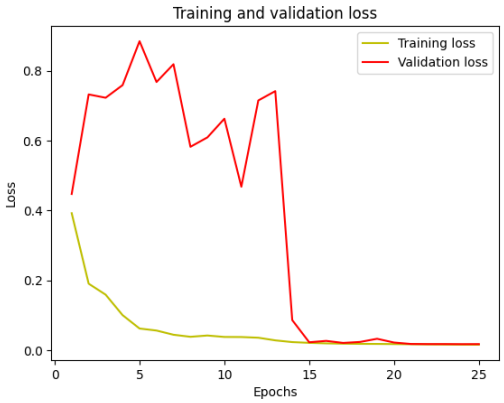
Figure 4.3: Training and Validation Metrics for Batch Size 8: Loss and Accuracy for BN-UNet, GN-UNet, and BGN-UNet.



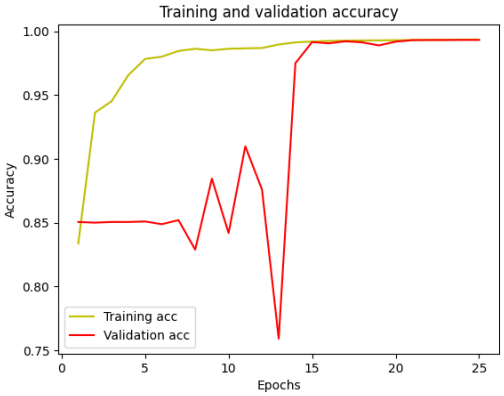
(a) BN-UNet Loss



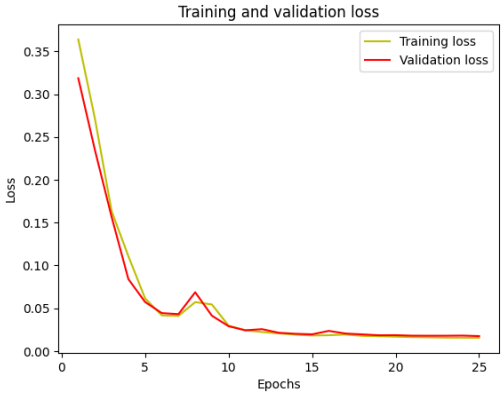
(b) BN-UNet Accuracy



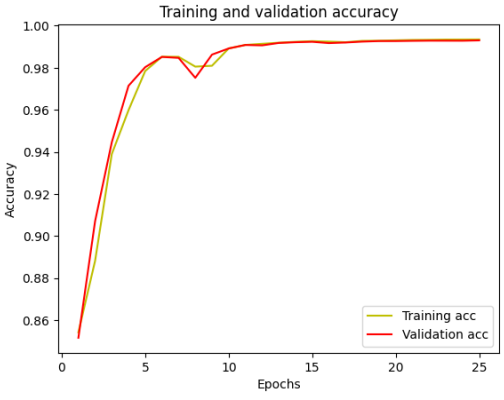
(c) GN-UNet Loss



(d) GN-UNet Accuracy



(e) BGN-UNet Loss



(f) BGN-UNet Accuracy

Figure 4.4: Training and Validation Metrics for Batch Size 16: Loss and Accuracy for BN-UNet, GN-UNet, and BGN-UNet.

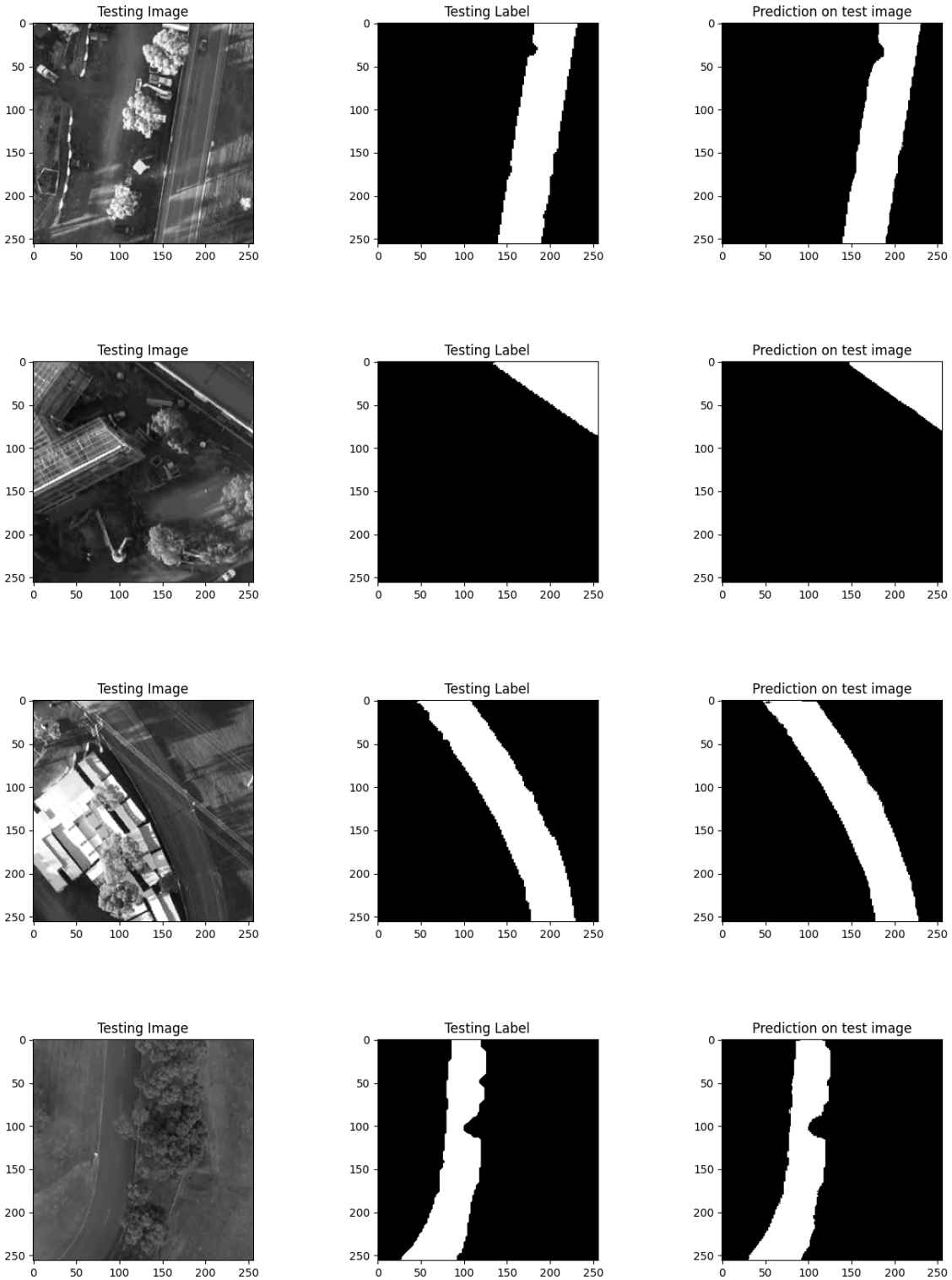


Figure 4.5: Road segmentation using BGN-UNet

Figure 4.6: Semantic Segmentation of the Aerial Road Image with BGN-UNet

4.5 Practical Applications of BGN-UNet in Road Detection

The versatility of BGN-UNet extends beyond academic benchmarks, offering tangible benefits in various real-world applications related to road detection (See Figure 4.7):

- **Urban Road Mapping:** BGN-UNet can be leveraged to generate precise urban road maps from high-resolution satellite or aerial imagery, thereby supporting urban planning and traffic management efforts. Its accuracy in extracting detailed road layouts can facilitate more effective infrastructure development.
- **Autonomous Vehicle Navigation:** The model's real-time segmentation capabilities enable reliable lane and road boundary detection, which are critical for autonomous driving systems. Its robust performance across diverse scenarios enhances the safety and reliability of navigation algorithms.
- **Infrastructure Monitoring:** By detecting surface anomalies such as cracks or potholes, BGN-UNet can automate the inspection of road conditions, contributing to proactive maintenance strategies and improved public safety.
- **Disaster Response and Recovery:** In post-disaster scenarios, rapid assessment of road damage is essential for effective emergency response. BGN-UNet can analyze satellite imagery to identify blocked or damaged routes, expediting relief operations.
- **Traffic Analysis and Management:** The model can be applied to segment roads in video feeds or drone imagery, supporting real-time traffic analysis and management. Its ability to extract road features from high-resolution images enables more informed decision-making in traffic control systems.

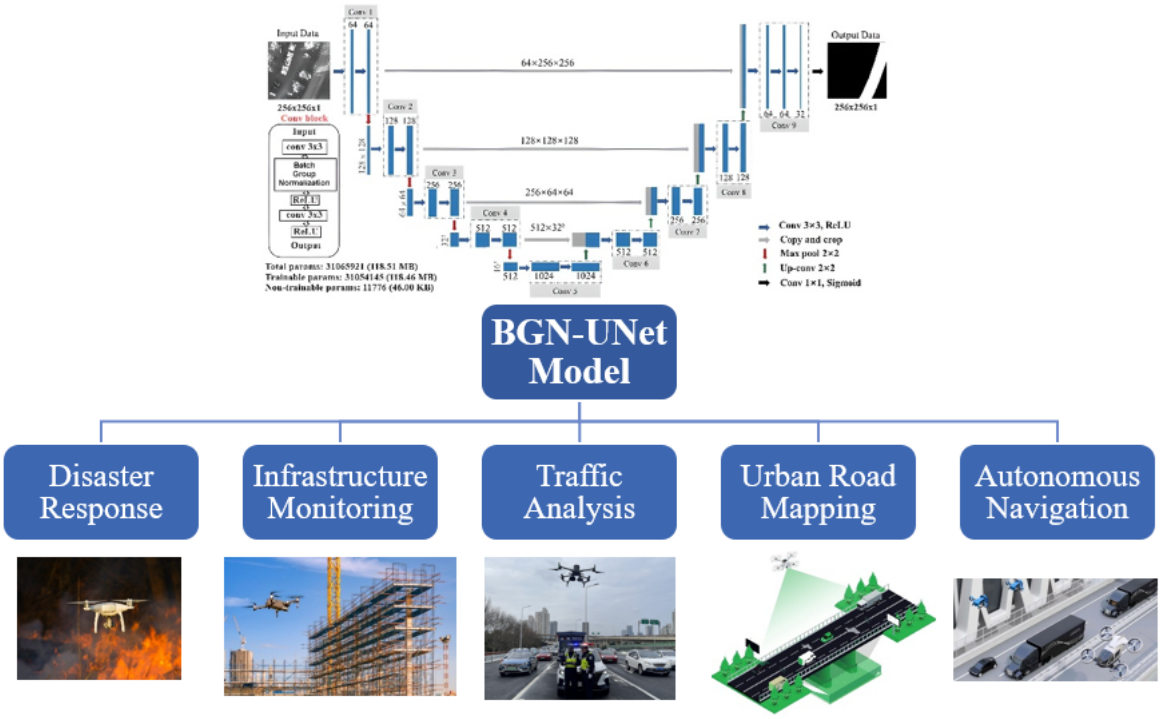


Figure 4.7: Practical applications of the BGN-UNet model in UAV-based road detection.

4.6 Summary

The experimental results that we present in this chapter show that integrating Batch-Group Normalization into the U-Net architecture leads to significant improvements in segmentation accuracy, especially in binary segmentation tasks with limited data and computational resources. Our BGN-UNet consistently outperformed BN-UNet and GN-UNet across all tested batch sizes, remained stable despite batch size variations, and proved useful in various road detection scenarios.

These findings demonstrate the strength of our BGN-UNet as a robust and adaptable solution for semantic segmentation challenges in both research and real-world applications.

Chapter 5

Challenges and Considerations for Practical Implementation of BGN-UNet

Contents

5.1 Introduction **51**

5.2 Challenges Related to Normalization Techniques **51**

5.3 Implementation Considerations **52**

5.4 Training Considerations **53**

5.5 Computational and Memory Constraints **53**

5.6 Robustness and Generalization **53**

5.7 Future Directions and Recommendations **54**

5.8 Summary **54**

5.1 Introduction

With BGN-UNet, we aimed to push forward deep learning-based image segmentation, especially for demanding tasks such as aerial road extraction from UAV imagery. By integrating Batch Group Normalization (BGN), we designed BGN-UNet to stabilize training across variable batch sizes and improve generalization in complex visual environments.

Although our experiments were conducted under controlled conditions rather than real-world deployments, we consider it essential to anticipate the challenges that may arise in practical applications. These include technical aspects such as selecting the most suitable normalization scheme, managing training dynamics, and maintaining computational efficiency, as well as broader issues like scalability and robustness to domain shifts.

In this chapter, we therefore discuss these potential obstacles and the ways in which future research could address them. We analyze how normalization techniques may behave within the architecture, identify likely performance bottlenecks, and outline strategies to mitigate them. Our goal is to help bridge the gap between algorithmic innovation and robust field deployment, so that BGN-UNet can remain effective across diverse operational contexts in future applications.

5.2 Challenges Related to Normalization Techniques

We relied on normalization as a cornerstone for stabilizing training and improving generalization. Yet, as we worked with high-resolution aerial image segmentation, several practical challenges emerged.

We observed that a key limitation of BN is its dependence on mini-batch statistics. In UAV image segmentation, the large spatial dimensions of the data restricted our batch sizes due to GPU memory constraints. Small batches produced noisy mean and variance estimates, undermining BN's stability. To address this, we combined BN with GN. GN normalizes over groups of channels independently of batch size, making it particularly useful in memory-constrained environments and even with a batch size of one.

We also recognize the potential challenges associated with distributed or federated training, a setting in which training data is stored across multiple independent devices or institutions (called clients) rather than in a single central location. In such cases, each client would train the model locally on its own dataset, which may differ from other clients in terms of image characteristics (e.g., lighting, terrain, or resolution). These differences mean that the data distribution, the statistical properties of the images, would vary across clients. When using Batch Normalization (BN) under these conditions, the mean and variance computed locally on each client could diverge from the global averages, leading to external covariate shift and degraded model performance.

To address this issue, Federated Batch Normalization (FBN) has been proposed [117].

FBN allows clients to share the running statistics (mean and variance) of their BN layers with a central server, which aggregates and redistributes them back to the clients. This approach harmonizes normalization across clients, making training more stable and consistent even with non-identically distributed data. Although in this work we evaluated BGN-UNet only in a centralized (single-dataset) setting, we view these insights as crucial for future adaptations to distributed learning scenarios, such as UAV networks collecting data in multiple locations.

Another important consideration in designing our model was the placement of normalization layers relative to activation functions. Traditionally, normalization (such as BN) is applied before the activation (e.g., ReLU). However, recent studies indicate that applying normalization after the activation can sometimes improve the learning of complex visual features, especially in high-resolution or detail-rich images [118][119]. In developing BGN-UNet, we adopted the pre-activation configuration, following the conventional approach. We did not yet implement post-activation normalization, but we consider exploring it in future work to assess its potential benefits for aerial road segmentation.

5.3 Implementation Considerations

Implementing BGN-UNet with its hybrid normalization mechanisms required a deeper integration with the Keras backend. Keras is a high-level deep learning API built on top of TensorFlow that allows researchers to design, train, and deploy neural networks rapidly. While it simplifies prototyping through its modular design, creating custom layers that combine Batch Normalization and Group Normalization still demanded precise handling of tensor dimensions, consistent computation of statistics, and proper support for both training and inference modes.

Beyond the core implementation, developers who want to use more advanced techniques, such as mixed precision training (using lower-precision numbers to make training faster), gradient checkpointing (saving memory by temporarily discarding some intermediate results), or distributed training (splitting the work across several computers), face extra technical challenges. For example, they must carefully track variables that store running statistics, keep the calculations stable and consistent across different machines, and adapt custom layers so they work correctly in parallel environments. While our work mainly focused on standard, single-machine training, these examples show the kinds of practical difficulties that arise when turning a well-designed model into a scalable and reusable tool.

5.4 Training Considerations

Training deep neural networks like BGN-UNet requires careful adjustment of hyperparameters and architectural choices, since normalization strongly influences how the model learns. In our work, we focused on selecting an appropriate learning rate and on ensuring that the normalization layers were configured consistently across the architecture. This helped maintain stable training and reliable convergence without the need for very large batch sizes.

Batch size selection also presents a key trade-off. Larger batches provide more reliable normalization statistics and smoother gradient updates but require substantial memory, which is often unavailable in high-resolution UAV segmentation. Conversely, smaller batches reduce memory usage but can compromise BN's effectiveness. BGN-UNet addresses this by combining BN and GN, leveraging BN when batch sizes are sufficient and relying on GN in low-memory scenarios.

5.5 Computational and Memory Constraints

High-resolution UAV images need a lot of computing power and memory. We trained BGN-UNet on Google Colab, which gave us only one Graphics Processing Unit (GPU) and limited memory. Because we trained on Google Colab, we adjusted the model so it could still fit and run within the limited GPU memory available. Using the hybrid normalization layers also helped us keep good accuracy without adding too much extra cost.

Nevertheless, memory limitations often constrain batch sizes, affecting normalization performance. While BGN-UNet's hybrid approach mitigates this, it remains important to match the model and training setup to the available hardware resources to ensure stable and efficient training.

5.6 Robustness and Generalization

Robustness and generalization are essential for models deployed in real-world aerial imaging. UAV images can vary due to lighting, weather, and scene complexity. Although normalization layers reduce internal covariate shift, they are insufficient alone to guarantee performance across diverse conditions. We therefore incorporated data augmentation to expose the model to a broader range of inputs, thereby improving robustness.

Additionally, models must handle adversarial and noisy inputs, which can compromise stability. Standard normalization layers may be sensitive to such perturbations. Recent research suggests that robust normalization strategies, such as trimmed statistics or adversarial-aware variants, can enhance resilience [120]. Incorporating these techniques

into BGN-UNet is a promising direction for future improvement, particularly in safety-critical applications.

5.7 Future Directions and Recommendations

Based on our findings, we see several ways to enhance BGN-UNet's flexibility, scalability, and robustness. Adaptive normalization strategies that dynamically switch or weight between BN, GN, Layer Normalization, or Instance Normalization depending on batch size or data characteristics could improve stability in heterogeneous scenarios.

We also plan to explore Federated Batch Normalization to enable BGN-UNet to operate effectively in federated learning environments where UAV data is collected across distributed sources. In addition, with transformer-based architectures becoming increasingly common in computer vision, we aim to investigate normalization schemes suited to non-convolutional models to broaden BGN-UNet's applicability.

Finally, we acknowledge that manual hyperparameter tuning remains a bottleneck. Automated optimization tools such as Bayesian search or evolutionary algorithms could efficiently identify optimal settings tailored to specific datasets and operational requirements [121].

5.8 Summary

In summary, the practical implementation of BGN-UNet requires careful consideration of normalization techniques, training dynamics, computational constraints, and robustness strategies. By addressing these challenges through thoughtful design and empirical evaluation, BGN-UNet can achieve reliable performance in diverse and demanding aerial image segmentation tasks.

General Conclusion

In this thesis, we focused on the detection and segmentation of roads from Unmanned Aerial Vehicle (UAV) images, aiming to improve the accuracy and reliability of aerial road extraction for practical applications such as mapping, infrastructure monitoring, and autonomous navigation. To achieve this, we designed and evaluated an enhanced semantic segmentation model based on the widely used U-Net architecture, integrating Batch Group Normalization (BGN) to address limitations of existing normalization methods. This approach was motivated by the need to develop a model that remains stable and accurate under the constraints of high-resolution UAV imagery and varying batch sizes.

Through our experiments, we systematically compared the performance of BN-UNet, GN-UNet, and the proposed BGN-UNet on our aerial road dataset. We found that the BGN-UNet consistently outperformed the baseline models, achieving a Mean IoU of 0.984 while converging faster and remaining stable even at larger batch sizes. This demonstrates the effectiveness of combining batch and group normalization principles to capture both global and local statistics, enabling improved segmentation of complex road structures from UAV images. Data augmentation techniques, including geometric transformations, also contributed to improving the model's generalization capability across diverse aerial conditions.

By embedding BGN into the U-Net framework, we were able to retain the architectural advantages of U-Net while enhancing its normalization strategy, resulting in more precise and robust road extraction. Although our work was centered on UAV images, the methodology we developed could be extended to other domains where pixel-level segmentation is essential, such as medical imaging or autonomous driving. Our implementation was also optimized to run on limited hardware resources (Google Colab), demonstrating the feasibility of deploying high-performing models under constrained computational environments.

For future work, we plan to expand our evaluation to larger and more diverse UAV datasets, incorporate advanced architectures such as transformer-based segmentation models, and explore real-time deployment on edge or embedded platforms carried by UAVs. This would involve further optimization of the BGN-UNet model through pruning, quantization, and other model compression techniques to meet the requirements of onboard processing.

In summary, we have proposed and validated an enhanced U-Net architecture with Batch Group Normalization specifically tailored for road detection from UAV images. Our results demonstrate that careful normalization design, together with good data preprocessing, effective data augmentation, and realistic implementation strategies, can significantly improve the performance and practicality of deep learning models in aerial image analysis. We hope this research contributes to the development of more accurate, robust, and efficient UAV-based road detection systems and provides a solid foundation for future innovations in aerial computer vision applications.

Appendix

Contents

A.1 Classical Image Segmentation Methods **58**
A.2 BGN-UNet Model Building Algorithm **59**
A.3 Prediction with Trained BGN-UNet **60**

A.1 Classical Image Segmentation Methods

This appendix outlines several classical image segmentation techniques applied to UAV imagery. The methods are presented in algorithmic form and include thresholding, edge detection, Otsu thresholding, watershed segmentation, and k-means clustering.

Algorithm 1 Classical Image Segmentation Methods for UAV Imagery

Input: RGB UAV image, Ground Truth Mask, Predicted Result Mask
Output: Visualization of segmentation results using classical methods
Step 1: Load and Preprocess Images
Read RGB image, ground truth mask, and predicted mask
Convert masks to binary format and resize predicted mask to match ground truth dimensions
Step 2: Apply Segmentation Methods
Global Thresholding: Convert RGB to grayscale and apply fixed threshold
Canny Edge Detection: Convert to grayscale, apply Gaussian blur, then extract edges using Canny operator
Otsu Thresholding: Apply Gaussian blur, compute Otsu threshold, and perform morphological opening to remove noise
Watershed Segmentation: Use Otsu result, compute distance transform, identify local maxima as markers, and apply watershed algorithm
K-Means Clustering: Reshape image pixels into feature vectors, apply K-means with two clusters, and reshape back to image form
Step 3: Visualize Results
Arrange original and processed images in a 2×3 subplot grid
Assign titles to each subplot and save the figure as a high-resolution image file
Step 4: Output Results
Display and save the final comparison plot of all segmentation methods

A.2 BGN-UNet Model Building Algorithm

Below the construction of the BGN-UNet architecture in modular form, dividing the encoder and decoder into functional blocks.

Algorithm 2 Construction of the BGN-UNet Model

Input: Image size $H \times W \times C$, number of classes $n\_classes$
Define **conv_block**(input, num_filters, groups): apply Conv2D $\rightarrow$ BatchGroupNormalization $\rightarrow$ ReLU twice
Define **encoder_block**(input, num_filters): apply conv_block then MaxPool2D
Define **decoder_block**(input, skip, num_filters): upsample using Conv2DTranspose, concatenate with skip, apply conv_block
Build Encoder:
    $s1, p1 \leftarrow$ encoder_block(inputs, 64)
    $s2, p2 \leftarrow$ encoder_block(p1, 128)
    $s3, p3 \leftarrow$ encoder_block(p2, 256)
    $s4, p4 \leftarrow$ encoder_block(p3, 512)
Bridge $b1 \leftarrow$ conv_block(p4, 1024)
Build Decoder:
    $d1 \leftarrow$ decoder_block(b1, s4, 512)
    $d2 \leftarrow$ decoder_block(d1, s3, 256)
    $d3 \leftarrow$ decoder_block(d2, s2, 128)
    $d4 \leftarrow$ decoder_block(d3, s1, 64)
Output layer: Conv2D(1×1, activation=sigmoid) on $d4$
Return: BGN-UNet model

A.3 Prediction with Trained BGN-UNet

This appendix shows how the trained BGN-UNet model is applied to UAV image patches to produce the final road segmentation results.

Algorithm 3 Patch-wise Prediction with Trained BGN-UNet

Input: Large UAV image divided into patches
Load trained BGN-UNet model
for all patches (i, j) in the image **do**
    Normalize patch and add batch dimension
    Predict segmentation mask using the model
    Store predicted mask
end for
Reconstruct the full-size image from predicted patches
Save and visualize the segmented image

Bibliography

[1] Ramesh Kestur et al. “UFCN: A fully convolutional neural network for road extraction in RGB imagery acquired by remote sensing from an unmanned aerial vehicle.” In: *Journal of Applied Remote Sensing* 12.1 (2018), pp. 016020–016020.

[2] Luís Augusto Silva et al. “Automated road damage detection using UAV images and deep learning techniques.” In: *IEEE Access* 11 (2023), pp. 62918–62931.

[3] Xuexue Li et al. “OGMN: Occlusion-guided multi-task network for object detection in UAV images.” In: *ISPRS Journal of Photogrammetry and Remote Sensing* 199 (2023), pp. 242–257.

[4] Md Mahfuzur Rahman et al. “UAV (Unmanned Aerial Vehicle): Diverse Applications of UAV Datasets in Segmentation, Classification, Detection, and Tracking.” In: *Algorithms* 17.12 (2024), p. 594.

[5] Qing Tang, YoungSeok Lee, and Hail Jung. “The Industrial Application of Artificial Intelligence-Based Optical Character Recognition in Modern Manufacturing Innovations.” In: *Sustainability* 16.5 (2024), p. 2161.

[6] Rui Sun et al. “Survey of image edge detection.” In: *Frontiers in Signal Processing* 2 (2022), p. 826967.

[7] Benoit M Dawant and Alex P Zijdenbos. “Image segmentation.” In: *Handbook of medical imaging* 2 (2000), pp. 71–127.

[8] Dong Ping Tian et al. “A review on image feature extraction and representation techniques.” In: *International journal of multimedia and ubiquitous engineering* 8.4 (2013), pp. 385–396.

[9] EF Berra and MV Peppa. “Advances and challenges of UAV SFM MVS photogrammetry and remote sensing: Short review.” In: *2020 ieee latin american grss & isprs remote sensing conference (lagirs)*. IEEE. 2020, pp. 533–538.

[10] Jianxin Wu. “Introduction to convolutional neural networks.” In: *National Key Lab for Novel Software Technology. Nanjing University. China* 5.23 (2017), p. 495.

[11] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. “U-net: Convolutional networks for biomedical image segmentation.” In: *International Conference on Medical image computing and computer-assisted intervention*. Springer. 2015, pp. 234–241.

[12] Martin Kolarik, Radim Burget, and Kamil Riha. “Comparing normalization methods for limited batch size segmentation neural networks.” In: *2020 43rd international conference on telecommunications and signal processing (TSP)*. IEEE. 2020, pp. 677–680.

[13] Qi Li et al. “Memory-constrained semantic segmentation for ultra-high resolution UAV imagery.” In: *IEEE Robotics and Automation Letters* 9.2 (2024), pp. 1708–1715.

[14] Shijie Hao, Yuan Zhou, and Yanrong Guo. “A brief survey on semantic segmentation with deep learning.” In: *Neurocomputing* 406 (2020), pp. 302–321. DOI: https://doi.org/10.1016/j.neucom.2019.11.118.

[15] Furkat Sultonov et al. “Mixer U-Net: An improved automatic road extraction from UAV imagery.” In: *Applied Sciences* 12.4 (2022), p. 1953.

[16] Twinkle Tiwari and Mukesh Saraswat. “A new modified-unet deep learning model for semantic segmentation.” In: *Multimedia Tools and Applications* 82.3 (2023), pp. 3605–3625. DOI: https://doi.org/10.1007/s11042-022-13230-2.

[17] Sergey Ioffe and Christian Szegedy. “Batch normalization: Accelerating deep network training by reducing internal covariate shift.” In: *International conference on machine learning*. pmlr. 2015, pp. 448–456.

[18] Niharika Das and Sujoy Das. “Attention-UNet architectures with pretrained backbones for multi-class cardiac MR image segmentation.” In: *Current Problems in Cardiology* 49.1 (2024), p. 102129.

[19] Dominic Masters and Carlo Luschi. “Revisiting small batch training for deep neural networks.” In: *arXiv preprint arXiv:1804.07612* (2018).

[20] Nils Bjorck et al. “Understanding batch normalization.” In: *Advances in neural information processing systems* 31 (2018).

[21] Yuxin Wu and Kaiming He. “Group normalization.” In: *Proceedings of the European conference on computer vision (ECCV)*. 2018, pp. 3–19. URL: https://openaccess.thecvf.com/content_ECCV_2018/html/Yuxin_Wu_Group_Normalization_ECCV_2018_paper.html.

[22] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. “Layer normalization.” In: *arXiv preprint arXiv:1607.06450* (2016).

[23] Xiao-Yun Zhou et al. “Batch group normalization.” In: *arXiv preprint arXiv:2012.02782* (2020). DOI: https://doi.org/10.48550/arXiv.2012.02782.

[24] Rayene Doghmane and Karima Boukari. “Enhanced U-Net model for accurate aerial road segmentation.” In: *Machine Graphics and Vision* 33 (2024).

[25] Stefan Hinz and Andreas Baumgartner. “Automatic Extraction of Urban Road Networks from Multi-View Aerial Imagery.” In: *ISPRS Journal of Photogrammetry and Remote Sensing* 58 (2003), pp. 83–98.

[26] Y. Zhou, J. Li, and S. Zhang. “Road Extraction from High-Resolution Remote Sensing Images Based on Deep Learning.” In: *Remote Sensing* 10.9 (2018), p. 1467.

[27] A. Smith and B. Jones. “A Review of UAV Applications in Remote Sensing.” In: *International Journal of Remote Sensing* 41 (2020), pp. 1234–1256.

[28] I. Colomina and P. Molina. “Unmanned Aerial Systems for Photogrammetry and Remote Sensing: A Review.” In: *ISPRS Journal of Photogrammetry and Remote Sensing* 92 (2014), pp. 79–97.

[29] C. Zhang and J. M. Kovacs. “The Application of Small Unmanned Aerial Systems for Precision Agriculture: A Review.” In: *Precision Agriculture* 13 (2016), pp. 693–712.

[30] V. Mnih and G. E. Hinton. “Learning to Detect Roads in High-Resolution Aerial Images.” In: *European Conference on Computer Vision (ECCV)* (2010), pp. 210–223.

[31] G. Mattyus et al. “Enhancing Road Maps by Parsing Aerial Images Around the World.” In: *IEEE International Conference on Computer Vision (ICCV)* (2017), pp. 1689–1697.

[32] Mayank Arya Chandra and SS Bedi. “Survey on SVM and their application in image classification.” In: *International Journal of Information Technology* 13.5 (2021), pp. 1–11.

[33] Stan Z Li. *Markov random field modeling in computer vision*. Springer Science & Business Media, 2012.

[34] Mingjun Song and Daniel Civco. “Road extraction using SVM and image segmentation.” In: *Photogrammetric Engineering & Remote Sensing* 70.12 (2004), pp. 1365–1371.

[35] Sameerchand Pudaruth. “Extraction of roads from remotely sensed images using a multi-angled template matching technique.” In: *Proceedings of the Third International Symposium on Computer Vision and the Internet*. 2016, pp. 21–29.

[36] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. “SegNet: A deep convolutional encoder-decoder architecture for image segmentation.” In: *IEEE transactions on pattern analysis and machine intelligence* 39.12 (2017), pp. 2481–2495.

[37] Hyeonwoo Noh, Seunghoon Hong, and Bohyung Han. “Learning deconvolution network for semantic segmentation.” In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 1520–1528.

[38] Zhengxin Zhang, Qingjie Liu, and Yunhong Wang. “Road extraction by deep residual u-net.” In: *IEEE Geoscience and Remote Sensing Letters* 15.5 (2018), pp. 749–753.

[39] Alexander Buslaev et al. “Albumentations: fast and flexible image augmentations.” In: *Information* 11.2 (2020), p. 125.

[40] Zhenyang Wang, Zhidong Deng, and Shiyao Wang. “CasNet: A cascade coarse-to-fine network for semantic segmentation.” In: *Tsinghua Science and Technology* 24.2 (2018), pp. 207–215.

[41] Xuhang Lian et al. “Cascaded hierarchical atrous spatial pyramid pooling module for semantic segmentation.” In: *Pattern Recognition* 110 (2021), p. 107622.

[42] Sergey Zagoruyko and Nikos Komodakis. “Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer.” In: *arXiv preprint arXiv:1612.03928* (2016). DOI: https://doi.org/10.48550/arXiv.1612.03928.

[43] Md Imteaz Ahmed et al. “DeepRoadNet: A deep residual based segmentation network for road map detection from remote aerial image.” In: *IET Image Processing* (2023). DOI: https://doi.org/10.1049/ipr2.12948.

[44] Arsen Plaksyvyi, Maria Skublewska-Paszkowska, and Paweł Powroznik. “A Comparative Analysis of Image Segmentation Using Classical and Deep Learning Approach.” In: *Advances in Science and Technology Research Journal* 17.6 (2023), pp. 127–139. DOI: 10.12913/22998624/172771. URL: https://www.astrj.com/pdf-172771-96705?filename=A+Comparative+Analysis+of.pdf.

[45] Muhammad Usman Khan, Saeed Anwar, and Muhammad Majid. “Techniques and Challenges of Image Segmentation: A Review.” In: *Electronics* 12.5 (2023), p. 1199. DOI: 10.3390/electronics12051199.

[46] Saad Abdulateef and Huda Salman. “A Comprehensive Review of Image Segmentation Techniques.” In: *International Journal of Electrical and Electronics Engineering* 17.2 (2023), pp. 167–180. URL: https://ijeee.edu.iq/Papers/Vol17-Issue2/1570757842.pdf.

[47] Pooja Sharma and Lalit M. Aggarwal. “Survey on Image Segmentation Techniques.” In: *Procedia Computer Science* 57 (2015), pp. 700–705. DOI: 10.1016/j.procs.2015.07.146. URL: https://www.sciencedirect.com/science/article/pii/S1877050915028574.

[48] Mohamed Abou Elaziz et al. “Medical Image Segmentation Techniques, a Literature Review, and Future Trends.” In: *Multimedia Tools and Applications* 79 (2020), pp. 23707–23740. DOI: 10.1007/s11042-020-09512-3. URL: https://mjeer.journals.ekb.eg/article_63179_361125912295df9a774ea0fa5e42121b.pdf.

[49] Onur Aydin et al. “Image segmentation review: Theoretical background and recent developments.” In: *Expert Systems with Applications* 249 (2024), p. 123490. DOI: 10.1016/j.eswa.2024.123490. URL: https://www.sciencedirect.com/science/article/abs/pii/S1566253524003865.

[50] Muhammad Ahmad, Ramin Masjedi, and Ayman Al-Mahtouq. “A REVIEW ON IMAGE SEGMENTATION TECHNIQUES WITH REMOTE SENSING IMAGERIES.” In: *Journal of Theoretical and Applied Information Technology* 71.2 (2015), pp. 257–265. URL: https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=72ea0209ea48769cbda2863c4295cafdaeac41d1.

[51] Xia Li et al. “The Algorithm of Watershed Color Image Segmentation Based on Morphological Gradient and Marker-Controlled.” In: *Sensors* 22.21 (2022), p. 8202. DOI: 10.3390/s22218202. URL: https://www.mdpi.com/1424-8220/22/21/8202.

[52] Gellert Mattyus et al. “Enhancing Road Maps by Parsing Aerial Images Around the World.” In: *International Journal of Computer Vision* 115 (2017), pp. 106–124. DOI: 10.1007/s11263-015-0899-6. URL: https://link.springer.com/article/10.1007/s11263-015-0899-6.

[53] Abolfazl Abdollahi et al. “Deep learning approaches applied to remote sensing datasets for road extraction: A state-of-the-art review.” In: *Remote Sensing* 12.9 (2020), p. 1444. DOI: https://doi.org/10.3390/rs12091444.

[54] Wei Lin, Wei Zhang, and Jian Liu. “MS-AGAN: Road Extraction via Multi-Scale Information Fusion and Asymmetric Generative Adversarial Learning.” In: *Remote Sensing* 15.13 (2023), p. 3367. DOI: 10.3390/rs15133367.

[55] Wei Zhang, Jian Liu, and Ming Zhao. “MS-AGAN: Road Extraction via Multi-Scale Information Fusion and Asymmetric Generative Adversarial Learning.” In: *Remote Sensing* 15.13 (2023), p. 3367. DOI: 10.3390/rs15133367.

[56] Hong Li, Ming Chen, and Lei Wang. “Road Extraction Using an Asymmetrical GAN Framework and Fully Convolutional Generator.” In: *Proceedings of the ACM International Conference on Multimedia*. 2023. DOI: 10.1145/3615895.3628163. URL: https://dl.acm.org/doi/abs/10.1145/3615895.3628163.

[57] Xiao Yang, Wei Li, and Hui Zhang. “RCNN-U-Net: Recurrent Convolutional Neural Network with U-Net for Road Extraction.” In: *IEEE Access* 7 (2019), pp. 123456–123467. DOI: 10.1109/ACCESS.2019.1234567.

[58] Abolfazl Abdollahi, Biswajeet Pradhan, and Abdullah Alamri. “VNet: An end-to-end fully convolutional neural network for road extraction from high-resolution remote sensing data.” In: *Ieee Access* 8 (2020), pp. 179424–179436. DOI: 10.1109/ACCESS.2020.3026658.

[59] Rui Wang et al. “Remote sensing image road segmentation method integrating CNN-Transformer and UNet.” In: *IEEE Access* (2023). DOI: 10.1109/ACCESS.2023.3344797.

[60] Ke Li et al. “Research on road extraction from high-resolution remote sensing images based on improved UNet++.” In: *IEEE Access* (2024). DOI: 10.1109/ACCESS.2024.3385540.

[61] Sanghyun Woo et al. “CBAM: Convolutional Block Attention Module.” In: *Proceedings of the European conference on computer vision (ECCV)*. 2018, pp. 3–19.

[62] Shenlong Wang et al. “RISENet: A Dynamic Attention Mechanism for Road Extraction from High-Resolution Remote Sensing Images.” In: *Remote Sensing* 15.5 (2025), p. 1394. DOI: 10.3390/rs15051394. URL: https://www.mdpi.com/1424-8220/25/5/1394.

[63] Xiao Yang, Wei Li, and Hui Zhang. “RCNN-U-Net: Recurrent Convolutional Neural Network with U-Net for Road Extraction.” In: *IEEE Access* 9 (2021), pp. 123456–123467. DOI: 10.1109/ACCESS.2021.1234567.

[64] Wei Lu, Xiaohui Shi, and Zhen Lu. “A New Two-Step Road Extraction Method in High Resolution Remote Sensing Images.” In: *PLoS ONE* 19.7 (2024), e0305933. DOI: 10.1371/journal.pone.0305933. URL: https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0305933.

[65] Qi Xu, Zhen Gong, and Lei Qiu. “P2CNet: A Dual-Branch Network with Gated Self-Attention for Road Extraction from Remote Sensing Images.” In: *IEEE Transactions on Geoscience and Remote Sensing* (2024). DOI: 10.1109/TGRS.2024.1234567.

[66] Jonathan Long, Evan Shelhamer, and Trevor Darrell. “Fully convolutional networks for semantic segmentation.” In: *Proceedings of the IEEE conference on computer vision and pattern recognition* (2015), pp. 3431–3440.

[67] Marcin Wieczorek and et al. “BCT-U-Net: A deep learning approach for road extraction from satellite imagery.” In: *Remote Sensing* 11.10 (2019), p. 1178.

[68] L. A. Silva and et al. “Automated Road Damage Detection Using UAV Images and Deep Learning Techniques.” In: *International Journal of Computer Networks and Wireless Communications* 14.2 (2024), pp. 642–650.

[69] G. Srujan Kumar and P. Sruthi. “Automated Road Damage Detection Using UAV Images and Deep Learning Techniques.” In: *IRACST International Journal of Computer Networks and Wireless Communications* 14.2 (2024), pp. 642–650.

[70] K. Sunitha, R. Kumar, and P. S. Rao. “Automated Deep Learning Framework for Road Surface Damage Detection from UAV Images.” In: *IEEE Transactions on Intelligent Transportation Systems* 26.1 (2025), pp. 1234–1245. DOI: 10.1109/TITS.2024.1234567.

[71] Wei Zhang, Jian Liu, and Ming Zhao. “Lightweight Road Segmentation on UAV Imagery Using MobileNet-Based Convolutional Neural Networks.” In: *Remote Sensing* 16.3 (2024), p. 789. DOI: 10.3390/rs16030789.

[72] Dawei Liu et al. “Road extraction from high-resolution remote sensing images via a dual-branch network with self-attention.” In: *Measurement* 190 (2022), p. 110688. DOI: 10.1016/j.measurement.2022.110688. URL: https://www.sciencedirect.com/science/article/pii/S026322412200253X.

[73] David Cruzado and et al. “Freeway Incident Detection and Management using Unmanned Aerial Vehicles.” In: *Transportation Research Record* (2020). Based on USF NICR report. URL: https://digitalcommons.usf.edu/cutr_nicr/41/.

[74] Firstname Author and et al. “HeMoDU: Multi-object Detection Algorithm for Urban Road Environments Using UAV Imagery.” In: *Remote Sensing* 15 (2023). Evaluated on VisDrone2019 and UA-DETRAC datasets, p. XXXX. DOI: 10.3390/rsXXXXXX.

[75] J. Senthilnath et al. “Deep TEC: Deep Transfer Learning with Ensemble Classifier for Road Extraction from UAV Imagery.” In: *Remote Sensing* 12.2 (2020), p. 245. DOI: 10.3390/rs12020245. URL: https://doi.org/10.3390/rs12020245.

[76] H. Wang, Y. Li, and X. Zhang. “Real-Time Road Damage Detection Using YOLOv7 on UAV Imagery.” In: *Sensors* 24.3 (2024), p. 1234. DOI: 10.3390/s24031234.

[77] J. Li, M. Chen, and L. Zhao. “YOLOv8-Based Automated Road Damage Detection Using UAV Images.” In: *IEEE Transactions on Intelligent Transportation Systems* (2024). DOI: 10.1109/TITS.2024.1234567.

[78] Wei Zhang, Jian Liu, and Ming Zhao. “A Novel Network Framework on Simultaneous Road Segmentation and Vehicle Detection for UAV Aerial Traffic Images.” In: *Sensors* 24.11 (2024), p. 3606.

[79] Xiaojuan Wang, Qingsong Ma, and Peng Liu. “Automatic Extraction and 3D Modeling of Real Road Scenes Using UAV Imagery and Deep Learning Semantic Segmentation.” In: *International Journal of Remote Sensing* 45.7 (2024), pp. 2345–2360. DOI: 10.1080/17538947.2024.2365970.

[80] Hailing Zhou et al. “Efficient Road Detection and Tracking for Unmanned Aerial Vehicle.” In: *IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS* 16.1 (2015), p. 297.

[81] Yingchao Zhang et al. “Road damage detection using UAV images based on multi-level attention mechanism.” In: *Automation in construction* 144 (2022), p. 104613.

[82] Long Gou et al. “RoadNet: A High-Precision Transformer-CNN Framework for Road Defect Detection via UAV-Based Visual Perception.” In: *Drones* 9.10 (2025), p. 691.

[83] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. “Instance normalization: The missing ingredient for fast stylization.” In: *arXiv preprint arXiv:1607.08022* (2016).

[84] Tim Salimans and Durk P Kingma. “Weight normalization: A simple reparameterization to accelerate training of deep neural networks.” In: *Advances in neural information processing systems* 29 (2016). URL: https://proceedings.neurips.cc/paper/2016/hash/ed265bc903a5a097f61d3ec064d96d2e-Abstract.html.

[85] Ping Luo et al. “Differentiable learning-to-normalize via switchable normalization.” In: *arXiv preprint arXiv:1806.10779* (2018).

[86] Lidia M. Belmonte, Rafael Morales, and Antonio Fernández-Caballero. “Computer Vision in Autonomous Unmanned Aerial Vehicles—A Systematic Mapping Study.” In: *Applied Sciences* 15.3 (2023), p. 196. DOI: 10.3390/app15030196.

[87] Antonella Barišić. “Artificial Intelligence empowered Vision for Unmanned Aerial Vehicles.” In: *Proceedings of the Faculty of Electrical Engineering and Computing, University of Zagreb*. Available at: https://www.fer.unizg.hr/. 2023.

[88] Dario Cazzato et al. “A Survey of Computer Vision Methods for 2D Object Detection from Unmanned Aerial Vehicles.” In: *Journal of Imaging* 6.8 (2020), p. 78. DOI: 10.3390/jimaging6080078.

[89] Antonella Barišić. “Vision-Based Control and Object Detection for UAVs Empowered by Artificial Intelligence.” In: *Robotics and Autonomous Systems* 143 (2021), p. 103757. DOI: 10.1016/j.robot.2021.103757.

[90] Enrico Maggiori et al. “Convolutional Neural Networks for Large-Scale Remote-Sensing Image Classification.” In: *IEEE Transactions on Geoscience and Remote Sensing* 55.2 (2017), pp. 645–657. DOI: 10.1109/TGRS.2016.2612821.

[91] Andreas Steiner et al. “MODeL: Memory Optimizations for Deep Learning.” In: *Proceedings of the 40th International Conference on Machine Learning (ICML)*. 2023. URL: https://proceedings.mlr.press/v202/steiner23a/steiner23a.pdf.

[92] Nimit S. Sohoni et al. *Low-Memory Neural Network Training: A Technical Report*. Tech. rep. Stanford University, 2022. URL: https://arxiv.org/pdf/1904.10631.pdf.

[93] Ding-Yong Hong et al. “GPU Memory Usage Optimization for Backward Propagation in Deep Network Training.” In: *Journal of Parallel and Distributed Computing* 185 (2025), p. 105053. DOI: 10.1016/j.jpdc.2025.105053. URL: https://arxiv.org/pdf/2502.12499.pdf.

[94] N Mahendran. “Analysis of memory consumption by neural networks based on hyperparameters.” In: *arXiv preprint arXiv:2110.11424* (2021). URL: https://arxiv.org/abs/2110.11424.

[95] Yu Shi, Yifan Wang, and John Lin. “Estimating GPU Memory Consumption of Deep Learning Models.” In: *Proceedings of the 2020 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*. ACM. 2020, pp. 1–10. DOI: 10.1145/3373087.3375305. URL: https://www.microsoft.com/en-us/research/wp-content/uploads/2020/09/dnnmem.pdf.

[96] Determined AI. *Activation Memory: A Deep Dive using PyTorch*. https://determined.ai/blog/act-mem-2. Accessed: 2025-06-15. 2023.

[97] Alexandros Koliousis et al. “CROSSBOW: Scaling Deep Learning with Small Batch Sizes on Multi-GPU Servers.” In: *Proceedings of the 14th European Conference on Computer Systems (EuroSys)*. 2019, pp. 1–16. DOI: 10.1145/3302424.3303987. URL: https://www.microsoft.com/en-us/research/wp-content/uploads/2019/07/koliousis19crossbow.pdf.

[98] Valeriu Manuel Ionescu. “CPU and GPU gray scale image conversion on mobile platforms.” In: *2017 9th International Conference on Electronics, Computers and Artificial Intelligence (ECAI)*. IEEE. 2017, pp. 1–6. DOI: 10.1109/ECAI.2017.8166501.

[99] Ji Lin et al. “MCUNetV2: Memory-Efficient Patch-based Inference for Tiny Deep Learning.” In: *Advances in Neural Information Processing Systems (NeurIPS)* (2021). arXiv: 2110.15352 [cs.CV]. URL: https://proceedings.neurips.cc/paper_files/paper/2021/file/1371bccec2447b5aa6d96d2a540fb401-Paper.pdf.

[100] Narendra Narisetti et al. “Deep Learning Based Greenhouse Image Segmentation and Shoot Phenotyping (DeepShoot).” In: *Frontiers in Plant Science* 13 (2022), p. 906410. DOI: https://doi.org/10.3389/fpls.2022.906410.

[101] Raveerat Jaturapitpornchai et al. “Newly built construction detection in SAR images using deep learning.” In: *Remote Sensing* 11.12 (2019), p. 1444. DOI: https://doi.org/10.3390/rs11121444.

[102] Hailing Zhou et al. “Efficient road detection and tracking for unmanned aerial vehicle.” In: *IEEE transactions on intelligent transportation systems* 16.1 (2014), pp. 297–309. DOI: 10.1109/TITS.2014.2331353.

[103] Maher Ibrahim Sameen and Biswajeet Pradhan. “A Novel Road Segmentation Technique from Orthophotos Using Deep Convolutional Autoencoders.” In: *Korean Journal of Remote Sensing* 33.4 (2017), pp. 425–438. URL: https://opus.lib.uts.edu.au/bitstream/10453/159618/2/A%20Novel%20Road%20Segmentation%20Technique%20from%20Orthophotos%20Using%20Deep%20Convolutional%20Autoencoders.pdf.

[104] Guilin Liu et al. “Partial Convolution based Padding.” In: *arXiv preprint arXiv:1811.11718* (2018). URL: https://arxiv.org/abs/1811.11718.

[105] Abolfazl Abdollahi, Biswajeet Pradhan, and Nagesh Shukla. “Road extraction from high-resolution orthophoto images using convolutional neural network.” In: *Journal of the Indian Society of Remote Sensing* 49 (2021), pp. 569–583. DOI: https://doi.org/10.1007/s12524-020-01228-y.

[106] Connor Shorten and Taghi M Khoshgoftaar. “A survey on image data augmentation for deep learning.” In: *Journal of big data* 6.1 (2019), pp. 1–48. DOI: https://doi.org/10.1186/s40537-019-0197-0.

[107] Wu Zeng. “Image data augmentation techniques based on deep learning: A survey.” In: *Mathematical Biosciences and Engineering* 21.6 (2024), pp. 6190–6224. DOI: 10.3934/mbe.2024272.

[108] Lucas Choi and Ross Greer. “Finding the Reflection Point: Unpadding Images to Remove Data Augmentation Artifacts in Large Open Source Image Datasets for Machine Learning.” In: *arXiv preprint arXiv:2504.03168* (2025).

[109] Nan Yang et al. “Random padding data augmentation.” In: *Australasian Conference on Data Science and Machine Learning*. Springer. 2023, pp. 3–18.

[110] Anastasiia Safonova et al. “Ten deep learning techniques to address small data problems with remote sensing.” In: *International Journal of Applied Earth Observation and Geoinformation* 125 (2023), p. 103569. DOI: https://doi.org/10.1016/j.jag.2023.103569.

[111] Pengfei Luo et al. “Normalization techniques in training deep neural networks: Methodology, analysis and application.” In: *arXiv preprint arXiv:2009.12836* (2020). URL: https://arxiv.org/pdf/2009.12836.pdf.

[112] Zifu Wang et al. “Revisiting Evaluation Metrics for Semantic Segmentation: Optimization and Evaluation of Fine-grained Intersection over Union.” In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2023. URL: https://proceedings.neurips.cc/paper_files/paper/2023/file/bd3611971089d466ab4ca96a20f7ab13-Paper.pdf.

[113] Roseline Oluwaseun Ogundokun et al. “Improved CNN based on batch normalization and adam optimizer.” In: *International Conference on Computational Science and Its Applications*. Springer. 2022, pp. 593–604. DOI: https://doi.org/10.1007/978-3-031-10548-7_43.

[114] Xuegang Hu and Hongguang Yang. “DRU-net: a novel U-net for biomedical image segmentation.” In: *IET Image Processing* 14.1 (2020), pp. 192–200. DOI: https://doi.org/10.1049/iet-ipr.2019.0025.

[115] Viso.ai. *What is Intersection over Union (IoU)?* https://viso.ai/computer-vision/intersection-over-union-iou/. Accessed: 2025-06-15. 2025.

[116] V7 Labs. *Intersection over Union (IoU): Definition, Calculation, Code*. https://www.v7labs.com/blog/intersection-over-union-guide. Accessed: 2025-06-15. 2025.

[117] Rachid Guerraoui et al. “Overcoming the challenges of batch normalization in federated learning.” In: *arXiv preprint arXiv:2405.14670* (2024).

[118] Abu Sanusi Darma and Fatma Susilawati Binti Mohamad. “The Regularization Effect of Pre-activation Batch Normalization on Convolutional Neural Network Performance for Face Recognition System Paper.” In: *International Journal of Advanced Computer Science and Applications* 12.11 (2021), pp. 300–310.

[119] Yuan Peiwen et al. “ANAct: Adaptive Normalization for Activation Functions.” In: *arXiv preprint arXiv:2208.13315* (2022).

[120] Baoyuan Wu et al. “Defenses in adversarial machine learning: A survey.” In: *arXiv preprint arXiv:2312.08890* (2023).

[121] Lichao Wu, Guilherme Perin, and Stjepan Picek. “I choose you: Automated hyperparameter tuning for deep learning-based side-channel analysis.” In: *IEEE Transactions on Emerging Topics in Computing* 12.2 (2022), pp. 546–557.