Research Square

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

Neurosymbolic Vision Mamba With Hypergraph Reasoning For Anatomically Consistent Diabetic Foot Ulcer Segmentation

Jan Adnan Farooq

janadnanfarooq@gmail.com

C. V. Raman Global University https://orcid.org/0009-0009-4985-4721

Shayam Ahmad

C. V. Raman Global University https://orcid.org/0009-0007-3648-4616

Haroon Iqbal

C. V. Raman Global University https://orcid.org/0009-0007-4133-1315

Research Article

Keywords: diabetic foot ulcer, wound segmentation, state space model, hypergraph neural network, neurosymbolic learning, test time augmentation

Posted Date: November 19th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-11718975/v1

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

Additional Declarations: The authors declare no competing interests.

NEUROSYMBOLIC VISION MAMBA WITH HYPERGRAPH REASONING FOR ANATOMICALLY CONSISTENT DIABETIC FOOT ULCER SEGMENTATION

A PROJECT REPORT

Submitted by

Haroon Iqbal (7th Sem) 2301020798

Shayam Ahmad (7th Sem) 2301020102

Jan Adnan Farooq (7th Sem) 2301020802

In partial fulfilment for the award of the degree of

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE & INFORMATION TECHNOLOGY



C.V. RAMAN GLOBAL UNIVERSITY

BHUBANESWAR - ODISHA - 752054

OCTOBER 2026



C.V. RAMAN GLOBAL UNIVERSITY

BHUBANESWAR - ODISHA - 752054

BONAFIDE CERTIFICATE

Certified that this project report "Neurosymbolic Vision Mamba with Hypergraph Reasoning for Anatomically Consistent Diabetic Foot Ulcer Segmentation" is the bonafide work of Haroon Iqbal, Shayam Ahmad and Jan Adnan Farooq, 7th Semester, Registration Nos. 2301020798, 2301020102, 2301020802 respectively, CGU Odisha, Bhubaneswar, who carried out the project under my supervision.

SUPERVISOR
Dr. Debendra Muduli
Head of the Department, CSE AIDS
C.V. Raman Global University

C.V. RAMAN GLOBAL UNIVERSITY

BHUBANESWAR - ODISHA - 752054

CERTIFICATE OF APPROVAL

This is to certify that we have examined the project entitled "Neurosymbolic Vision Mamba with Hypergraph Reasoning for Anatomically Consistent Diabetic Foot Ulcer Segmentation", the bonafide work submitted by Haroon Iqbal, Shayam Ahmad and Jan Adnan Farooq, 7th Semester, Registration Nos. 2301020798, 2301020102, 2301020802 respectively, CGU Odisha, Bhubaneswar.

We hereby accord our approval of it as a major project work carried out and presented in a manner required for its acceptance for the partial fulfilment for the Bachelor Degree of Computer Science and Information Technology for which it has been submitted. This approval does not necessarily endorse or accept every statement made, opinion expressed, or conclusions drawn as recorded in this major project; it only signifies the acceptance of the major project for the purpose it has been submitted.

SUPERVISOR
Dr. Debendra Muduli

DECLARATION

We declare that this project report titled “Neurosymbolic Vision Mamba with Hypergraph Reasoning for Anatomically Consistent Diabetic Foot Ulcer Segmentation”, submitted in partial fulfilment of the degree of B.Tech in Computer Science and Information Technology, is a record of original work carried out by us under the supervision of Dr. Debendra Muduli, and has not formed the basis for the award of any other degree or diploma, in this or any other Institution or University. In keeping with the ethical practice in reporting scientific information, due acknowledgements have been made wherever the findings of others have been cited.

Haroon Iqbal (Reg. No. 2301020798)
Shayam Ahmad (Reg. No. 2301020102)
Jan Adnan Farooq (Reg. No. 2301020802)
Bhubaneswar - 752054
October 2026

ACKNOWLEDGEMENTS

We express our sincere gratitude to our project supervisor, Dr. Debendra Muduli, for his guidance, advice and support throughout this project. His comments on the direction of the work and on the presentation of this report were of great help to us.

Haroon Iqbal
Shayam Ahmad
Jan Adnan Farooq

ABSTRACT

Diabetic foot ulcers are chronic wounds whose size and shape must be assessed repeatedly during treatment, and outlining them by hand is slow and varies between observers. This project built and compared four deep learning models that segment ulcer regions in clinical photographs. Each model combines an image encoder, a Mamba inspired gated context block, a hypergraph module over feature nodes and a convolutional decoder. The models differ in the encoder (ResNet 50 and ConvNeXt B with ImageNet weights, and two networks trained from scratch), in how nodes and hyperedges are constructed, in the loss function, and in whether a structural rule about wound bed and wound edge regions is applied.

The models were trained under one protocol on 1,370 images pooled from the Foot Ulcer Segmentation Challenge (FUSC) and Medetec collections (913 training, 228 validation and 229 test images) and evaluated with Dice, IoU, precision, recall and HD95, using exponential moving average weights and six fold test time augmentation. On the 168 test images that contain a wound, Model 2 reached a Dice of 0.9367 and Model 1 reached 0.9365, a difference far inside their bootstrap intervals; Model 4 reached 0.9007 and Model 3 reached 0.8759.

A review of the code showed that the evaluation is easier than whole photograph segmentation, because images were cropped around the reference mask, and that the 200 FUSC test photographs, whose masks the organisers withhold, were treated as empty masks. No ablation or baseline was run, so the contribution of the individual components is not established. The scores should be read within this protocol. The work is an exploratory architectural study and not a clinical tool.

Keywords: diabetic foot ulcer, wound segmentation, state space model, hypergraph neural network, neurosymbolic learning, test time augmentation

TABLE OF CONTENTS

DESCRIPTION **PAGE NUMBER**

BONAFIDE CERTIFICATE ii
CERTIFICATE OF APPROVAL iii
DECLARATION iv
ACKNOWLEDGEMENTS v
ABSTRACT vi
LIST OF FIGURES xii
LIST OF TABLES xiv
ABBREVIATIONS / NOTATIONS / NOMENCLATURE xvi
CHAPTER 1 INTRODUCTION 1
1.1 Background 1
1.2 Diabetic Foot Ulcers 1
1.3 Clinical Importance of Wound Assessment 2
1.4 Medical Image Segmentation 2
1.5 Challenges in Diabetic Foot Ulcer Segmentation 2
1.6 Role of Deep Learning in Wound Segmentation 3
1.7 Limitations of Conventional CNN Based Segmentation 3
1.8 Long Range Context in Medical Images 4
1.9 Vision Mamba and State Space Models 4
1.10 Hypergraph Reasoning for Medical Images 5
1.11 Neurosymbolic Learning and Anatomical Knowledge 5
1.12 Problem Statement 5
1.13 Motivation 6
1.14 Research Gap 6
1.15 Aim of the Project 6
1.16 Objectives 7
1.17 Research Questions 7
1.18 Scope of the Work 8
1.19 Contributions 8
1.20 Organization of the Thesis 9
CHAPTER 2 LITERATURE SURVEY 10
2.1 Introduction 10
2.2 Diabetic Foot Ulcer Image Analysis 10
2.3 Traditional Wound Segmentation 11
2.4 CNN Based Medical Image Segmentation 11
2.5 U Net and Encoder Decoder Architectures 12
2.6 Attention Based Medical Image Segmentation 12
2.7 Transformer Based Medical Image Segmentation 12
2.8 State Space Models 13
2.9 Mamba 13

2.10 Vision Mamba for Medical Image Analysis 14
2.11 Graph Neural Networks 14
2.12 Hypergraph Neural Networks 14
2.13 Neurosymbolic AI 15
2.14 Anatomical Priors in Medical Image Segmentation 15
2.15 Existing Diabetic Foot Ulcer Segmentation Methods 16
2.16 FUSC and Other Wound Segmentation Benchmarks 17
2.17 Segmentation Evaluation Metrics 18
2.18 Generalization and Cross Dataset Challenges 19
2.19 Comparative Analysis of Existing Research 19
2.20 Research Gaps 20
2.21 Positioning of the Present Work 21
CHAPTER 3 METHODOLOGY 23
3.1 Proposed System Overview 23
3.2 Overall Processing Pipeline 24
3.3 Dataset Sources 25
3.4 FUSC Dataset 26
3.5 Medetec Dataset 26
3.6 Combined Dataset 26
3.7 Dataset Structure 27
3.8 Dataset Exploration 27
3.9 Data Splitting Strategy 27
3.10 Preprocessing 28
3.10.1 Image Loading 29
3.10.2 Mask Loading 29
3.10.3 Foreground Cropping 29
3.10.4 Image Resizing 30
3.10.5 Image Normalisation 30
3.10.6 Training Augmentation 30
3.10.7 Mask Preserving Augmentation 30
3.11 Data Integrity and Label Handling 31
3.12 Model Development Framework 31
3.13 Common Architecture 32
3.14 Vision Mamba Bottleneck 33
3.15 Selective State Space Model (as implemented) 33
3.16 Hypergraph Representation 34
3.17 Hypergraph Node Construction 35
3.18 Anatomical Knowledge Representation 36
3.19 Neurosymbolic Fusion 37
3.20 Model 1: ResNet 50 + Vision Mamba + Hyperedge Attention 37
3.21 Model 2: ConvNeXt B + Vision Mamba + Dynamic Hypergraph 40
3.22 Model 3: U Net + Vision Mamba + Symbolic Hypergraph 42

3.23 Model 4: Multi Granularity Hypergraph 44
3.24 Loss Functions 48
3.24.1 Dice Loss 48
3.24.2 Binary Cross Entropy 48
3.24.3 Focal Loss 48
3.24.4 Lovász Loss 49
3.24.5 Boundary Dice Loss 49
3.24.6 Deep Supervision Loss 49
3.25 Optimization 49
3.26 Learning Rate Scheduling 50
3.27 Exponential Moving Average 50
3.28 Gradient Clipping 50
3.29 Early Stopping 51
3.30 Test Time Augmentation 51
3.31 Prediction Procedure 51
3.32 Evaluation Metrics 52
3.32.1 Dice Coefficient 52
3.32.2 Intersection over Union 52
3.32.3 Precision 52
3.32.4 Recall 52
3.32.5 Hausdorff Distance at the 95th Percentile 53
3.33 Model Complexity 53
3.34 Inference Time 54
3.35 Explainability and Architectural Analysis 54
3.36 Experimental Environment 55
3.37 Data Leakage Considerations 55
CHAPTER 4 EXPERIMENTAL RESULTS AND ANALYSIS 58
4.1 Experimental Overview 58
4.2 Dataset Statistics 58
4.3 Dataset Split 59
4.4 Sample Image and Mask Analysis 60
4.5 Training Behaviour 60
4.6 Model 1 Results 62
4.7 Model 2 Results 63
4.8 Model 3 Results 63
4.9 Model 4 Results 64
4.10 Comparative Model Performance 64
4.11 Dice Analysis 65
4.12 IoU Analysis 67
4.13 Precision Analysis 68
4.14 Recall Analysis 68
4.15 HD95 Boundary Analysis 68

4.16 Confusion and Error Region Analysis 69
4.17 Training Convergence 69
4.18 Validation Performance 69
4.19 Test Time Augmentation Analysis 70
4.20 Empty Mask Test Image Analysis 70
4.21 Confidence Interval Analysis 71
4.22 Model Complexity Analysis 72
4.23 Computational Performance 73
4.24 Hypergraph Analysis 73
4.25 Symbolic Fusion Analysis 75
4.26 Model 3 Fusion Gate Analysis 76
4.27 Model 4 Multi Granularity Analysis 78
4.28 Qualitative Segmentation Results 80
4.29 False Positive Analysis 84
4.30 False Negative Analysis 85
4.31 Architectural Comparison and Component Analysis 86
4.32 Comparison with Existing Research 86
4.33 Discussion 87
4.34 Limitations 88
4.35 Threats to Validity 88
CHAPTER 5 CONCLUSION AND FUTURE WORK 90
5.1 Summary of the Work 90
5.2 Major Findings 90
5.3 Achievement of Objectives 92
5.4 Answers to Research Questions 92
5.5 Contributions 93
5.6 Practical Significance 93
5.7 Limitations 94
5.8 Future Work 94
5.9 Final Conclusion 95
REFERENCES 96
Appendix A Dataset Description 99
Appendix B Dataset Statistics 100
Appendix C Model Architecture Details 101
Appendix D Hyperparameter Configuration 103
Appendix E Mathematical Formulations 105
Appendix F Training Records 106
Appendix G Test Set Results 107
Appendix H Error Evidence 108
Appendix I Hypergraph Visualisations 109
Appendix J Symbolic Fusion Visualisations 110
Appendix K Multi Granularity Analysis 111

Appendix L Computational Performance 112
Appendix M Implementation Environment 113
Appendix N Source Code 114

LIST OF FIGURES

FIGURE	TITLE	PAGE NUMBER
3.1	Overall system architecture	25
3.2	Training and inference pipeline	25
3.3	Two stage stratified split of the pooled dataset	28
3.4	Preprocessing workflow	31
3.5	Common architecture of the four models	32
3.6	Vision Mamba bottleneck and the block it contains	33
3.7	Hypergraph construction	36
3.8	Model 1 architecture	39
3.9	Model 1 hyperedge types on a wound image (notebook output)	39
3.10	Model 2 architecture	41
3.11	Model 2 node fusion and dynamic hypergraph pipeline (notebook output)	42
3.12	Model 3 architecture	43
3.13	Model 3 symbolic and visual hyperedges over the four region types (notebook output)	44
3.14	Model 4 architecture	46
3.15	Model 4 multi granularity hyperedge construction (notebook output)	46
3.16	Hierarchical message passing in Model 4	47
4.1	Number of images in each source folder	59
4.2	Sample images, reference masks and overlays (notebook output)	60
4.3	Training curves of Model 1 (top) and Model 2 (bottom)	61
4.4	Training curves of Model 3 (top) and Model 4 (bottom)	62
4.5	Test set metrics (EMA weights, six fold TTA) of Models 1 to 4 with 95% bootstrap intervals	65
4.6	Per image Dice, IoU and HD95 distributions on the lesion containing test images (Models 1 to 4, top to bottom)	67
4.7	Model 2 node branch analysis, incidence matrix and node degree distribution (notebook output)	75
4.8	Model 1 symbolic gate values per edge node and their distribution (notebook output)	76
4.9	Model 3 fusion parameter α over the epochs and its final value per region	77
4.10	Model 3 wound bed prior gate values	78
4.11	Model 3 symbolic and visual adjacency matrices for one sample	78
4.12	Model 4 node activation maps at the fine, mid and coarse levels for four samples (notebook output)	79
4.13	Model 4 cross scale gate activations (notebook output)	79
4.14	Model 4 hyperedge types on a representative wound image (notebook output)	80
4.15	Model 1: input image, reference mask, prediction and difference map for six test images (TP green, FP red, FN blue)	81
4.16	Model 2: input image, reference mask, prediction and difference map for six test images (TP green, FP red, FN blue)	82
4.17	Model 3: input image, reference mask, prediction and difference map for six test images (TP green, FP red, FN blue)	83

FIGURE	TITLE	PAGE NUMBER
4.18	Model 4: input image, reference mask, prediction and difference map for six test images (TP green, FP red, FN blue)	84
A.1	Model 1 hyperedge construction schematic (notebook output)	109
A.2	Model 2 input image with reference mask, graph nodes and dynamic hyperedges (notebook output)	109
A.3	Model 3 hypergraph nodes by region type and symbolic hyperedges (notebook output)	110
A.4	Model 3 symbolic and visual hyperedges for one sample (notebook output)	110
A.5	Model 4 multi granularity node positions and cross scale hyperedges (notebook output)	111

LIST OF TABLES

TABLE	TITLE	PAGE NUMBER
1.1	Challenges in diabetic foot ulcer segmentation	3
2.1	Summary of the studies reviewed	19
2.2	Comparison of wound segmentation approaches (qualitative, from Sections 2.3 and 2.15)	20
2.3	Comparison of CNN, transformer, Mamba and graph based approaches	20
2.4	Identified research gaps	21
3.1	Dataset composition as read by the notebooks	26
3.2	Dataset split produced by the implementation	28
3.3	Preprocessing operations (all four models)	29
3.4	Training augmentation operations	30
3.5	Placement of the Vision Mamba block	33
3.6	Node index ranges used as anatomical groups	36
3.7	Model architecture comparison	47
3.8	Model specific loss functions	47
3.9	Model specific input and node configuration	48
3.10	Common training configuration	50
3.11	Evaluation metrics	52
3.12	Architectural analyses implemented	54
3.13	Hardware and software environment	55
4.1	Dataset distribution	59
4.2	Training and validation summary (validation scores computed on all 228 validation images with the ordinary, non EMA weights)	61
4.3	Test set performance on the 168 lesion containing images (EMA weights, six fold TTA; mean, standard deviation, minimum, maximum over images)	64
4.4	Model comparison (rank 1 is best on that metric)	64
4.5	Validation Dice at selected epochs (read from the training logs)	69
4.6	Lesion only against full test set scores	70
4.7	Bootstrap 95% confidence intervals (168 lesion containing test images)	71
4.8	Parameter counts by module (millions)	72
4.9	FLOPs for one 512 × 512 input (thop, multiply accumulates doubled)	72
4.10	Inference latency in the experimental environment (single image, no TTA, Kaggle GPU)	73
4.11	Hypergraph characteristics	74
4.12	Gate and fusion values	75
4.13	Error evidence available from the metrics and prediction grids	85
4.14	Reported results in the literature and in this project (context only; no ranking is implied)	87
A.1	Dataset composition as read by the notebooks	99
A.2	Dataset distribution (images and reference masks)	99
B.1	Dataset split produced by the implementation	100
C.1	Model architecture comparison	101

TABLE	TITLE	PAGE NUMBER
C.2	Placement of the Vision Mamba block	101
C.3	Model specific input and node configuration	101
C.4	Node index ranges used as anatomical groups	102
D.1	Common training configuration	103
D.2	Model specific loss functions	103
D.3	Preprocessing operations (all four models)	103
D.4	Training augmentation operations	104
E.1	Equations used in the thesis	105
E.2	Evaluation metrics as implemented	105
F.1	Training and validation summary	106
F.2	Validation Dice at selected epochs (read from the training logs)	106
G.1	Test set performance on the 168 lesion containing images (EMA weights, six fold TTA)	107
G.2	Bootstrap 95% confidence intervals (168 lesion containing test images)	107
G.3	Lesion only against full test set scores	107
H.1	Error evidence available from the metrics and prediction grids	108
L.1	Parameter counts by module (millions)	112
L.2	FLOPs for one 512 × 512 input	112
L.3	Inference latency in the experimental environment (single image, no TTA)	112
M.1	Hardware and software environment	113

ABBREVIATIONS / NOTATIONS / NOMENCLATURE

Term	Meaning
CNN	Convolutional neural network
DFU	Diabetic foot ulcer
FUSC / FUSeg	Foot Ulcer Segmentation Challenge / dataset
Medetec	Medetec wound image collection
SSM	State space model
HGNN	Hypergraph neural network
FPN	Feature pyramid network
BCE	Binary cross entropy
IoU	Intersection over union
HD95	Hausdorff distance at the 95th percentile (pixels)
TP, FP, FN	True positive, false positive and false negative pixel counts
TTA	Test time augmentation
EMA	Exponential moving average of weights
FLOPs	Floating point operations (multiply accumulates doubled in this report)
GFLOPs	10^9 floating point operations
AdamW	Adam optimiser with decoupled weight decay
GPU	Graphics processing unit
M1, M4	Models 1 to 4 of this project
N	Number of hypergraph nodes
H	Hypergraph adjacency matrix
τ	Similarity threshold used to form hyperedges
σ	Sigmoid function
α_r, θ_r	Fusion weight and its parameter for region r (Model 3)
L	Length of the token sequence in the context block

CHAPTER 1
INTRODUCTION

1.1 Background

Measuring a wound is part of treating it. Clinicians follow whether an ulcer is shrinking or growing by recording its size and outline at each visit, and they use these records to decide whether the current treatment is working. When the records are made by hand, with a ruler or by tracing the border on a transparent sheet, they take time and they depend on who does the measuring. Digital photographs offer a way to make the process more consistent, provided the wound region in the photograph can be outlined reliably. Outlining a region of interest in an image, pixel by pixel, is called segmentation, and doing it automatically with a trained computer model is the subject of this thesis.

The project described here builds and compares four deep learning models that segment foot ulcers in clinical photographs. The models combine a convolutional image encoder with three further ideas that have been proposed for medical imaging: a context block modelled on Mamba, which lets every part of the image feature map draw on the rest at low computational cost; a hypergraph module, which lets groups of image regions exchange information; and, in some of the models, a fixed structural rule inspired by the way a wound is described clinically. The sections below explain why each of these was of interest, what the technical problem is, and what this thesis does and does not set out to show.

1.2 Diabetic Foot Ulcers

A diabetic foot ulcer (DFU) is an open sore on the foot of a person with diabetes. The Foot Ulcer Segmentation Challenge paper describes DFUs as a common and serious complication of diabetes that can lead to amputation of the lower limb, and presents objective wound measurement as important for following the healing process [18]. Ulcers differ greatly in size, depth, location on the foot and appearance. They may show red granulation tissue, yellowish slough, dark necrotic tissue, dry callus around the edge, or a mixture, and the surrounding skin may be inflamed or dry.

1.3 Clinical Importance of Wound Assessment

Assessment of a chronic wound normally includes its dimensions, the state of the tissue in the wound bed, the condition of the wound edge and of the surrounding skin, and signs of infection. Of these, the area and outline are the quantities most directly obtainable from a photograph. Because they are recorded repeatedly over weeks, small systematic errors in the outline accumulate into errors in the apparent rate of healing, and differences between observers make the series harder to interpret. These observations motivate research on automatic segmentation as a way to produce an outline that is reproducible, in the sense that the same image always gives the same outline.

The thesis keeps one distinction in view throughout. A segmentation model outputs an outline. It does not diagnose, it does not grade severity, and it does not recommend treatment. Any clinical use would need validation on data from the intended setting, under clinical supervision, which this project did not carry out.

1.4 Medical Image Segmentation

In semantic segmentation, a model assigns a class to every pixel. For wound images the problem is binary: each pixel is either wound or not wound. The model is trained on images paired with reference masks drawn by human annotators, and its output is compared with the reference masks of images it

has not seen. Training uses a loss function that measures the disagreement between prediction and reference, and the quality of the final model is described by overlap measures such as the Dice coefficient and the intersection over union, and by boundary measures such as the 95th percentile Hausdorff distance (Section 2.18) [29, 32].

1.5 Challenges in Diabetic Foot Ulcer Segmentation

Table 1.1 summarises the difficulties that make DFU segmentation harder than segmentation of structures with a regular appearance. The challenges are drawn from the studies reviewed in Chapter 2 and from the images used in this project (Figure 4.2).

Table 1.1: Challenges in diabetic foot ulcer segmentation

Challenge	Why it matters
Wide variation in size, shape and colour	A fixed template does not fit; the model must handle both small ulcers and ulcers covering much of the image
Irregular, often unclear borders	The outline separating wound from skin is the quantity of interest and is hard to place consistently, also for human annotators
Uncontrolled imaging conditions	Photographs differ in lighting, camera distance, angle, background and resolution
Small labelled datasets	Annotation needs expert time, so datasets are small compared with natural image datasets
Several tissue types inside one wound	Different colours and textures inside the wound make it look like several regions
Variation between annotators	The reference mask itself is an approximate record of an uncertain boundary
Availability of labels	Even public challenges may withhold test labels, which limits direct comparison between studies [18]

1.6 Role of Deep Learning in Wound Segmentation

Convolutional neural networks have become the standard tool for medical image segmentation [1]. For wound images, studies have used fully convolutional networks, U Net variants with pretrained encoders, ensembles of several networks, and attention augmented decoders [19, 20, 21]. On the FUSeg challenge, networks with ImageNet pretrained encoders, ensembling and test time augmentation reached scores of 88.80% and 89.23% Dice on the official test set [20, 21]. Pretrained encoders are widely used because labelled wound images are few. A model that has already learned to represent edges, colours and textures from a large natural image collection needs less wound data to adapt [35, 36]. This theme returns in Chapter 4, where the two models with pretrained encoders score higher than the two without.

1.7 Limitations of Conventional CNN Based Segmentation

A convolutional layer combines the values in a small neighbourhood. A deep stack of layers enlarges the region of the image each output depends on, but information from distant parts of the image reaches a given position only after passing through many layers and tends to be weak. Chen and colleagues give this as the reason for adding a transformer to the U Net: convolution is local, and a network that is purely convolutional models long range dependency only in a limited way [5]. In wound images this matters because the identity of a region often depends on its surroundings. A reddish patch is part of the wound if it is enclosed by a border and surrounded by intact skin, and is a flushed area of healthy skin if it is not. Whether the limitation of local receptive fields is the main source of error in DFU segmentation has not been established by the experiments in this thesis, since no purely convolutional counterpart of the proposed models was trained.

1.8 Long Range Context in Medical Images

Transformers give every patch of the image direct access to every other patch through self attention, and they have been combined with U Net in medical segmentation [5, 6]. The price is that the cost of attention grows with the square of the number of patches [7]. State space models offer a different route to long range context, in which information is carried along a sequence by a hidden state at a cost proportional to the length of the sequence [8, 9]. This is the motivation for the Mamba style context block used in this project.

1.9 Vision Mamba and State Space Models

Mamba lets the parameters of a state space model depend on the input, so that the model can choose what information to carry forward, and computes it with an efficient scan [9]. For images, the feature map has to be turned into a sequence, and Vision Mamba and VMamba do this by scanning in two or four directions [10, 11]. VM UNet and U Mamba brought these ideas to medical image segmentation [12, 13]. In this project the context block is a simplified form: it keeps the outer structure of a Mamba layer but mixes information along a single raster ordered sequence with a gated cumulative average and not with a state space recurrence (Section 3.15). The term “Vision Mamba” is used in the thesis for the role of the block, and its limits are stated where it is described.

1.10 Hypergraph Reasoning for Medical Images

A graph represents relations between pairs of items. A hypergraph allows one relation to join many items, and hypergraph neural networks pass features between nodes and the groups they belong to [15, 16]. For an image, nodes can be regions of the feature map, and a hyperedge can connect regions that share a property such as similar colour or belonging to the same part of the wound. The appeal for wound images is that the relation between the wound bed, the wound edge and the surrounding skin involves several regions together and not just pairs. Whether this is useful in practice depends on how the nodes and hyperedges are built, which is why the four models of this project use different constructions (Chapter 3), and on whether the hypergraph output influences the segmentation strongly, which Chapter 4 examines.

1.11 Neurosymbolic Learning and Anatomical Knowledge

Neurosymbolic approaches combine learned neural components with explicit symbolic structure [17]. In wound segmentation the symbolic element considered here is the clinical description of a wound as a bed surrounded by an edge and then by perilesional skin (Section 2.14). In the models of this project the rule is implemented as fixed groups of nodes with prescribed connections and a gating layer. This is a modest form of the idea: the rule fixes how information may flow between node groups, but the groups are assigned by index and are not aligned to the image (Section 3.18), and the experiments do not test whether removing the rule changes the results. The thesis therefore treats the symbolic component as an exploratory design element.

1.12 Problem Statement

Given a colour photograph of a foot ulcer, produce a binary mask of the ulcer region. The problem in this thesis is to design several deep learning architectures for this task that combine a convolutional encoder, a Mamba style context block, hypergraph reasoning and, optionally, a fixed anatomical structure prior; to train and evaluate them under the same protocol on a pooled public dataset; and to analyse how they differ in accuracy, boundary quality and computational cost, and what their internal gating and graph quantities show.

1.13 Motivation

Three considerations motivated the work. First, manual measurement of wounds is slow and observer dependent, and automatic outlining of wounds is an active area of research with public benchmarks [18]. Second, context modelling and relational reasoning methods have been proposed for other medical imaging tasks [5, 13, 15], and it was of interest to see how designs of this kind behave on wound photographs. Third, the explicit description of a wound's parts in clinical practice suggested adding structural knowledge to a network, and gate values and graph statistics offer a way to look inside such a model.

1.14 Research Gap

The gaps that follow from the literature of Chapter 2 are the following. Published results on DFU segmentation come from different test sets, and the official FUSeg test labels are not public, so methods evaluated on subsets of the public data cannot be ranked against leaderboard scores [18, 20, 21]. Among the wound segmentation studies reviewed, explicit structural prior knowledge about the wound was not found, and no study combining a state space type context block, a hypergraph module and a structural prior for DFU segmentation was found in the limited search carried out for this thesis. That search was not systematic, and the second gap in particular is stated with caution: Mamba based models for DFU images exist , and the absence of a combined design from a limited search is not proof that none has been published. The thesis therefore presents its work as an exploration and not as the filling of a verified gap.

1.15 Aim of the Project

The aim is to build and compare four deep learning architectures for automated diabetic foot ulcer segmentation that combine a convolutional encoder, a Mamba style context block, hypergraph reasoning and, in some of them, a fixed structural prior; and to report honestly what the experiments show about their accuracy, boundary behaviour, computational cost and internal gating.

1.16 Objectives

1. To develop an automated diabetic foot ulcer segmentation framework using deep learning.
2. To incorporate a Mamba style context block into the segmentation pipeline for global context modelling.
3. To investigate hypergraph based reasoning on node features derived from wound images.
4. To incorporate a fixed structural prior, inspired by wound anatomy, into selected architectures.
5. To develop and compare four architectural variants under one training and evaluation protocol.
6. To evaluate the models with region overlap and boundary based metrics.
7. To analyse the computational characteristics of the models: parameters, floating point operations where measured, and inference time in the experimental environment.
8. To analyse the recorded gate values and graph structure where the implementation allows.
9. To identify the strengths and limitations of each design and of the experimental procedure.

1.17 Research Questions

RQ1. How accurately do the four architectures segment diabetic foot ulcer regions in the pooled dataset used in this project?

RQ2. How do the four architectures differ in Dice, IoU, precision, recall and HD95?

RQ3. What do the recorded properties of the hypergraph constructions and of the symbolic or gating modules show about the behaviour of the models?

RQ4. What computational trade offs exist among the four architectures in parameter count, FLOPs where measured, inference latency and segmentation performance?

RQ5. What does the experimental procedure itself, in particular the data preparation, allow the results to show, and what does it not allow?

These questions can be answered from the experiments. No question asks whether any component improves performance over a baseline, because no baseline or ablation experiment was run. RQ5 was added to the questions listed in the original plan because the analysis of the code showed that data preparation choices affect how far the results can be interpreted (Sections 3.10.3, 3.11 and 4.20).

1.18 Scope of the Work

The work covers binary segmentation of foot ulcers in colour photographs; four architectures implemented in PyTorch; a pooled dataset of 1,370 images from the Foot Ulcer Segmentation Challenge and Medetec collections; and evaluation on a held out part of this pooled set. It does not cover diagnosis, severity grading, tissue type segmentation, detection of ulcers in uncropped photographs, conversion of pixel measurements to physical units, deployment, or clinical validation. The experiments used one training run per model, so seed to seed variation was not measured, and no controlled ablation or baseline was run.

1.19 Contributions

The contributions of the work, stated as far as the evidence allows, are the following.

1. Implementation of four segmentation architectures that combine a convolutional encoder, a Mamba style context block and hypergraph reasoning with different node constructions, hyperedge definitions, decoders and losses, and one of which uses a learned region wise fusion parameter.
2. A comparison of the four architectures under one protocol, with bootstrap confidence intervals, and a report of their parameter counts, FLOPs where measured, and latency.
3. A record of the internal quantities of the models (gate values, graph statistics, parameter shares), with an assessment of what they do and do not show.
4. A critical account of the data pipeline, which identifies the inclusion of images without available reference masks, the use of the reference mask in cropping, the actual split proportions, and the behaviour of the metrics on empty masks, together with their effect on how the results should be interpreted.

The thesis does not claim a new state of the art result, a new method that outperforms published ones, or clinical effectiveness.

1.20 Organization of the Thesis

Chapter 2 reviews the literature on DFU image analysis, segmentation architectures, state space models, hypergraph networks and neurosymbolic methods, and on datasets and metrics. Chapter 3

describes the dataset, preprocessing, the four architectures, the training procedure and the evaluation metrics, as implemented. Chapter 4 reports and analyses the experimental results, including computational cost, internal quantity analysis, limitations and threats to validity. Chapter 5 summarises the findings, answers the research questions, and proposes future work. The references and appendices follow.

CHAPTER 2
LITERATURE SURVEY

2.1 Introduction

This chapter describes what other researchers have done on the problem addressed in this project. It follows the development of the field in the order in which the ideas appeared: convolutional encoder decoder networks, attention mechanisms, transformers, state space models and Mamba, graph and hypergraph neural networks, and neurosymbolic approaches, and it then turns to the work on diabetic foot ulcer (DFU) images and the datasets and metrics used to evaluate it. The present project is not described here; it is placed against this background in Section 2.22.

Two cautions apply throughout. First, published segmentation scores cannot be compared without checking which dataset, which test set and which evaluation protocol each paper used; the chapter points out several cases in which numbers that look comparable are not. Second, a short list of studies is described in detail and the others are mentioned only where they help to explain a theme. The aim is to show how the field developed and where the open questions are, not to list every paper.

2.2 Diabetic Foot Ulcer Image Analysis

A diabetic foot ulcer is an open wound on the foot of a person with diabetes. The Foot Ulcer Segmentation Challenge paper describes the condition as a common and serious complication that can lead to lower limb amputation, and describes the need for objective measurement of wound area to follow healing [18]. In image based work the tasks can be grouped as follows.

- **Classification** of an image or image region as ulcer or healthy skin, or by type or severity.
- **Detection** of ulcers with bounding boxes. The Diabetic Foot Ulcer Challenge datasets and the evaluation of detection methods by Yap and colleagues belong here [28].
- **Segmentation** of the wound area, the task of this project, in which every pixel is labelled as wound or not wound.
- **Tissue segmentation**, in which the wound is divided into tissue types such as granulation, slough and necrosis, studied by Ramachandram and colleagues on mobile device photographs [23] and by Dhar and colleagues in the DFUTissue dataset (Section 2.16).

Wound segmentation matters because the area and outline of the wound are the basis of most measurements used to follow healing. The photographs are taken in everyday clinical conditions rather than in a controlled laboratory, which is the source of most of the technical difficulty described in the next sections.

Table 1.1 (Chapter 1) collects the challenges; they are listed here only as far as the literature below depends on them: large variation in wound size, shape and colour; irregular and often unclear borders; changing background, lighting and camera distance; small datasets; and annotations that vary between annotators.

2.3 Traditional Wound Segmentation

Before deep learning, wound segmentation relied on hand designed image processing. Typical methods used colour space thresholding, region growing, active contours and clustering, sometimes combined with a classifier trained on hand chosen colour and texture features. The studies cited by the deep learning papers of Section 2.15 describe these methods as dependent on the choice of features and on lighting conditions, and several of them required user interaction such as a seed point or a

manually removed background. Scebba and colleagues, whose work is described in Section 2.15, point out that earlier DFU segmentation methods were semi automatic and relied on user interaction [22].

2.4 CNN Based Medical Image Segmentation

Fully convolutional networks replaced hand designed features with learned ones by predicting a label for every pixel from convolutional features. For medical images the encoder decoder form became standard: an encoder reduces spatial size while increasing the number of channels, so that deeper layers see larger parts of the image, and a decoder restores the spatial size. Pretraining the encoder on ImageNet is a common way to compensate for small medical datasets; ResNet [35] and later ConvNeXt [36] are widely used as encoders, and both are used in this project. The feature pyramid network [37] is a decoder style design in which maps from several encoder depths are combined top down; it is also used in one model here.

A property shared by convolutional networks is that each output depends on a limited neighbourhood of the input, even though the neighbourhood grows with depth. This is why many later designs add a mechanism for gathering information from the whole image.

2.5 U Net and Encoder Decoder Architectures

U Net, introduced by Ronneberger, Fischer and Brox in 2015 [1], connects each encoder stage to the decoder stage of the same resolution by a skip connection, so the decoder receives both coarse semantic information and fine spatial detail. It was designed for small biomedical datasets and became the base design for many later models. UNet++ [2] replaced the plain skip connections with nested and dense ones to reduce the gap between encoder and decoder features. U Net style decoders with skip connections are used by three of the four models of this project, and the fourth uses the feature pyramid design.

2.6 Attention Based Medical Image Segmentation

Attention mechanisms let a network weight some features or positions more than others. In Attention U Net, Oktay and colleagues placed attention gates in the skip connections, so that irrelevant regions in the encoder features are suppressed before they are passed on [3]; Schlemper and colleagues developed the gated attention mechanism for medical images in a related journal paper [4]. For wound images, FUSegNet adds spatial and channel attention modules (parallel scSE) inside the decoder [21]. These mechanisms improve how features are weighted, but they still operate on local neighbourhoods, which led to the interest in transformers.

2.7 Transformer Based Medical Image Segmentation

The Vision Transformer [7] divides an image into patches and applies self attention among all patches, so each patch can use information from every other patch. TransUNet, proposed by Chen and colleagues, passes CNN feature maps through a transformer encoder to obtain global context and then uses a U Net decoder with skip connections to restore detail [5]. The authors describe the motivation as the limited ability of convolution to model long range dependency and the weak localisation of pure transformers. Swin Unet replaced convolutions by shifted window transformer blocks in both encoder and decoder [6]. Self attention has a cost that grows with the square of the number of patches, which limits the resolution at which it can be used without further measures, and transformer models usually need large pretraining datasets. These two points are the usual motivation given for looking at alternatives such as state space models.

2.8 State Space Models

A state space model describes how a hidden state changes along a sequence under the influence of the input, and produces an output from the state. The structured state space sequence model (S4) of Gu, Goel and Ré made this form practical for long sequences by choosing the state matrix so that the computation could be done efficiently [8]. Because the cost grows linearly with the length of the sequence, state space models were proposed as an alternative to attention for long sequences. In the basic form the parameters do not depend on the input, which limits the ability to select what to remember.

2.9 Mamba

Mamba, by Gu and Dao, made the state space parameters depend on the input, so the model can decide at each position what to keep and what to ignore, and used a hardware aware parallel scan to keep the computation linear in sequence length [9]. It was introduced for language and other one dimensional sequences. Applying it to images needs a way to turn two dimensional feature maps into sequences. Vision Mamba (Zhu and colleagues) used bidirectional scanning of image patches [10], and VMamba (Liu and colleagues) used a four direction scan over the feature map [11]. In both, scanning in several directions is the means by which the one dimensional model covers a two dimensional image.

The distinction matters for this project. The block implemented in this project keeps the outer form of a Mamba layer but replaces the selective scan with a gated cumulative average along a single raster ordered sequence (Section 3.15). It is therefore a simplified relative of Mamba and not an implementation of it, and the comparison with the literature of this section should be read in that light.

2.10 Vision Mamba for Medical Image Analysis

U Mamba combined convolutional blocks with state space blocks in a U shaped network for biomedical segmentation [12]. VM UNet, by Ruan, Li and Xiang, built a U shaped network entirely from visual state space blocks derived from VMamba and evaluated it on skin lesion datasets (ISIC17, ISIC18) and the Synapse multi organ dataset; the authors present it as the first purely state space segmentation model for medical images and as a baseline for later work [13]. A number of further Mamba based medical segmentation models followed. For wound images, the authors of the FUTransUNet paper list a lightweight Mamba based DFU segmentation model, “DFU MambaLiteUNet”, among their references [24]. The use of Mamba type models on DFU images is therefore not new in itself, and the thesis does not claim it as new.

2.11 Graph Neural Networks

Graph neural networks operate on nodes connected by edges and update each node’s feature from its neighbours. The graph convolutional network of Kipf and Welling is the standard formulation [14]. In imaging, nodes can be pixels, superpixels, image patches or regions, and edges can express spatial adjacency or feature similarity. The common advantage cited is that relations between regions that are far apart in the image can be represented directly. The limitation is that an edge connects only two nodes, so a relation that involves several regions at once has to be approximated by many pairwise edges.

2.12 Hypergraph Neural Networks

A hyperedge can join any number of nodes. Feng and colleagues defined hypergraph neural networks, in which features are propagated from nodes to hyperedges and back using an incidence matrix and node and hyperedge degrees [15]. Antelmi and colleagues surveyed representation learning on hypergraphs [16]. In image work, hyperedges are commonly built from nearest neighbours in feature space, so one hyperedge links a group of similar regions. In this thesis the claims made about medical use of hypergraph networks are limited to the statement that they are an established way of representing group relations and that their use in image analysis is documented in survey work [16].

2.13 Neurosymbolic AI

Neurosymbolic AI combines neural networks, which learn from data, with symbolic representations, which express explicit rules and structure. d'Avila Garcez and Lamb describe it as an attempt to combine the learning ability of neural networks with the reasoning and explanation ability of symbolic methods [17]. In image segmentation the usual forms are constraints in the loss function, post processing rules, and architecture modules that encode known relations. In the project's models, the rules are structural: a wound edge group of nodes is connected to a wound bed group of nodes (Section 3.18). This is a weak form of the idea, since the rule is a fixed pattern of connections and not logical inference, and the thesis uses the term in that limited sense.

2.14 Anatomical Priors in Medical Image Segmentation

Segmentation errors in medical images are often anatomically implausible, for example fragmented regions or regions in the wrong place. Priors about shape, topology or the arrangement of structures are therefore added to the loss, to the architecture or to post processing. For wounds, the usual clinical description is a wound bed surrounded by a wound edge and by perilesional skin. How strongly a network benefits from such a prior depends on whether it can be tied to the image content. In the project the groups are assigned by node index and not by image location (Section 3.18), which means that the prior is not anatomical in the sense used by the shape prior literature, where the prior is aligned to the image.

2.15 Existing Diabetic Foot Ulcer Segmentation Methods

The studies below were checked against their primary or index pages.

Wang and colleagues, FUSeg (2024) [18]. The paper describes the MICCAI 2021 Foot Ulcer Segmentation Challenge and its dataset of 1,210 foot ulcer photographs collected over two years from 889 patients, divided into training, validation and test sets. The paper reports that the test annotations were withheld by the organisers and that the challenge remained open as a benchmark afterwards. Limitation relevant here: the test labels are not public, so work that uses only the public data cannot be scored on the official test set.

Wang and colleagues, chronic wound dataset (2020) [19]. An earlier dataset of 1,010 foot ulcer images from 889 patients with masks for all images, which the FUSeg dataset extends [20]. Evaluated with MobileNetV2 based segmentation and tested on Medetec images as an additional test of generalisation.

Scebba and colleagues, Detect and Segment (2022) [22]. The authors first detect the wound position with one network, isolate the wound region from the background, and then segment inside the isolated region with a second network. They evaluated it on a DFU dataset and four further datasets with other wound types. They report that the Matthews correlation coefficient rose from 0.29 when the segmentation was computed on the full image to 0.85 when detection and segmentation were

combined, and that the approach allowed training with much less data. Relevance: segmentation inside a region already known to contain the wound is a much easier problem than segmentation of a full photograph, which is also the situation in this project's evaluation (Section 3.10.3).

Mahbod, Schaefer, Ecker and Ellinger, ensemble of CNNs (ICPR 2022) [20]. An ensemble of a LinkNet and a U Net, each with a pretrained EfficientNet encoder. The encoders were further trained on the Medetec dataset, and five fold cross validation, test time augmentation and result fusion were used. The paper reports data based Dice scores of 92.07% on the chronic wound dataset and 88.80% on the FUSeg challenge test set. Limitation relevant here: the two numbers refer to different test sets, and the lower one is on the official hidden test set.

Dhar and colleagues, FUSegNet (2024) [21]. An encoder decoder with a pretrained EfficientNet b7 encoder and a decoder module that combines spatial and channel attention (parallel scSE). Reported data based Dice of 92.70% on the chronic wound dataset and 89.23% in the FUSeg challenge with the cross validated variant x FUSegNet, which the authors state placed first on the challenge leaderboard. Again the two numbers belong to different test sets.

Asare and colleagues, FUTransUNet GradCAM (2025, preprint) [24]. A transformer U Net hybrid for foot ulcer segmentation, trained and tested on FUSeg, with Grad CAM visualisations used to explain predictions.

Other recent studies (Alabdulhafith and colleagues, a ResNet encoder U Net for chronic wound monitoring [25]; Hamrani and colleagues, a label free approach based on diffusion models [26]; Zhou and colleagues, a diagnostic pipeline combining segmentation and classification of diabetic foot [27]). No results from these three studies are quoted in this thesis.

2.16 FUSC and Other Wound Segmentation Benchmarks

FUSeg (FUSC). Described in Section 2.15 [18]. Its three parts have the following availability, which matters for this project: 810 training and 200 validation images with public masks, and 200 test images whose masks are held by the organisers.

Chronic wound dataset. 1,010 images with masks, from the earlier work of Wang and colleagues [19, 20].

Medetec. A collection of wound photographs from the Medetec wound database; its images are used for further pretraining in Mahbod and colleagues' work and as an additional test set in the work of Wang and colleagues [19, 20]. The collection contains several wound types and not only diabetic foot ulcers.

DFUTissue. A dataset for tissue segmentation in DFUs created by Dhar and colleagues [21].

Other benchmarks. The DFUC challenge datasets address detection [28].

2.17 Segmentation Evaluation Metrics

Overlap measures. The Dice coefficient, also called the F1 score for pixels, and the intersection over union (Jaccard index) compare predicted and reference regions; precision and recall separate over- from under segmentation. Dice was also adopted as a loss function by Milletari and colleagues [29].

Boundary measures. The Hausdorff distance compares the boundaries of two shapes [32]. Because the maximum is sensitive to single outlying points, the 95th percentile of boundary distances (HD95) is commonly reported instead. Aydin and colleagues show that the average Hausdorff distance can hide errors when used to rank methods [34], and Taha and Hanbury give guidance on choosing metrics for medical segmentation and analyse their properties [33]. These sources support two practical points that apply to this project: results depend on how empty masks and undefined distances are handled, and a single metric should not be used alone.

Losses. Focal loss reduces the weight of well classified pixels [30], and the Lovász hinge is a surrogate for the Jaccard loss [31]. Both are used in this project and are discussed in Section 3.24.

2.18 Generalization and Cross Dataset Challenges

Models trained on one collection of images often lose accuracy on images from another source, because of differences in camera, lighting, patient population and annotation style. Scebba and colleagues tested their approach on four additional datasets with other wound types for this reason [22], and Wang and colleagues tested on Medetec images [19]. The unavailability of the FUSeg test labels also means that results from public data alone cannot be compared directly with leaderboard values (Section 2.16). The project paper refers to a cross dataset benchmark for this point, but the citation could not be verified, so the chapter rests on the cited studies above.

2.19 Comparative Analysis of Existing Research

Table 2.1: Summary of the studies reviewed

Study	Task / data	Method	Reported result	Limitation relevant to this project
Wang et al. 2024 [18]	FUSeg challenge and dataset (1,210 images, 889 patients)	Dataset and challenge description	,	Test masks withheld
Scebba et al. 2022 [22]	DFU images plus four other wound datasets	Detect wound, isolate, then segment	MCC 0.29 (full image) → 0.85 (detect + segment)	Different metric; shows importance of localisation
Mahbod et al. 2022 [20]	Chronic wound dataset; FUSeg test	Ensemble LinkNet + U Net, pretrained, TTA	Dice 92.07% (chronic wound); 88.80% (FUSeg test)	Two different test sets
Dhar et al. 2024 [21]	Chronic wound dataset; FUSeg test	Encoder decoder, EfficientNet b7, parallel scSE	Dice 92.70% (chronic wound); 89.23% (FUSeg test, x FUSegNet)	Two different test sets
Chen et al. 2021 [5]	Multi organ and cardiac segmentation	CNN + transformer encoder, U Net decoder	, (not general wound data)	Not evaluated on wounds
Ruan et al. 2024 [13]	ISIC17, ISIC18, Synapse	Pure visual state space U Net	,	Not evaluated on wounds
Asare et al. 2025 [24]	FUSeg	Transformer U Net, Grad CAM	Not quoted here	Preprint

Table 2.2: Comparison of wound segmentation approaches (qualitative, from Sections 2.3 and 2.15)

Approach	Typical strength	Typical weakness
Classical image processing	No training data needed	Sensitive to lighting and background; user interaction
CNN encoder decoder (U Net, LinkNet) with pretraining	Strong with small data	Local receptive field
CNN with attention modules	Better feature weighting	Still local
Hybrid CNN transformer	Global context	Cost grows with number of patches; needs pretraining
Detect then segment pipelines	Large gain from localisation	Two models; errors of the detector propagate

Table 2.3: Comparison of CNN, transformer, Mamba and graph based approaches

Family	Global context	Cost with sequence length	Explicit relations between regions	Typical medical imaging use
CNN	Grows with depth	Linear in pixels	No	Dominant baseline
Transformer	Direct (all patches)	Quadratic	No	Hybrid encoders [5, 6]
Mamba / state space	Through sequence scan	Linear	No	U shaped models [12, 13]
Graph	Through edges	Depends on graph	Pairwise	Region reasoning [14]
Hypergraph	Through hyperedges	Depends on hypergraph	Group	Surveyed in [16]

Comparing the results. The Dice values in the wound segmentation papers of Section 2.15 lie between 88.80% and 92.70%, but they come from two different test sets. The two Mahbod and Dhar values that are higher (92.07% and 92.70%) are for the chronic wound dataset; the values on the official FUSeg test set are 88.80% and 89.23%. The project paper's comparison table lists the higher values under FUSeg and adds two further sources (labelled Huttunen and Wang) that give a score identical to FUSegNet's 92.70%, without a verified source. That table is therefore not reproduced in this thesis (Section 4.33).

2.20 Research Gaps

The following gaps follow from the studies reviewed above. They are limited to what could be checked in the literature search that was done for this chapter; this search was not a systematic review.

1. **Comparability.** Results on DFU segmentation come from different datasets and test sets, and the official FUSeg test labels are not public. A method evaluated on a random subset of the public data cannot be ranked against leaderboard scores.
2. **Context versus localisation.** Detect then segment work shows that localisation strongly changes measured performance [22]. Papers do not always state whether images are cropped, so it is unclear how often evaluation is on full photographs.
3. **Explicit structure.** Among the wound segmentation studies reviewed, the main means of improving results are pretrained encoders, attention modules, ensembling and hybrid transformer designs. Explicit structural prior knowledge about the wound (bed, edge, surrounding skin) was not found in them. This is an observation about the studies reviewed here, not a claim about the whole literature.
4. **State space and relational models on wounds.** Mamba based DFU models exist [24], but a combination of a state space type context block, hypergraph reasoning and a structural prior for DFU segmentation was not found in the search performed. Absence from the limited search carried out here is not proof that no such work exists.
5. **Interpretability.** Explanations in the wound segmentation papers reviewed are mostly post hoc (for example Grad CAM [24]); architectural quantities such as gate values are rarely analysed.

Table 2.4: Identified research gaps

Gap	Evidence from the literature	Addressed in this project?
Different test sets limit comparison	[18, 20, 21]	Not addressed; the project uses its own split
Effect of localisation	[22]	Not addressed; the project crops using the mask and acknowledges this
Structural priors in DFU segmentation	none found in reviewed studies	Explored through fixed node groups
Mamba + hypergraph + prior on wounds	not found in limited search	Explored in four architectures
Architectural interpretability	[24] shows post hoc only	Gate and graph statistics recorded

2.21 Positioning of the Present Work

This project builds four segmentation models that combine a pretrained or from scratch encoder, a simplified Mamba style context block, a hypergraph module over feature nodes, and in some models a fixed structural prior, and compares them on a pooled dataset of two public sources. It is an exploratory architectural study. It does not claim to exceed published methods, because its test set, input preparation and metrics differ from theirs; it does not contain a controlled baseline or ablation; and it does not claim a clinical result. What it adds to the literature above is a documented implementation and a comparison of four designs under one protocol, together with measurements of what the relational and gating components did, and a record of the practical problems found in the data pipeline.

CHAPTER 3
METHODOLOGY

3.1 Proposed System Overview

This chapter describes what was implemented in the four project notebooks. Four segmentation networks were built and trained on the same combined wound image collection, using the same optimiser settings, the same training schedule and the same evaluation code. They share a general layout: an image encoder produces feature maps, a context block mixes information across the whole feature map, a small set of "nodes" is taken from the features and passed through a hypergraph module, and a convolutional decoder turns the result into a binary wound mask. What differs between the models is the encoder, the way nodes are chosen, the way hyperedges are defined, and how anatomical knowledge about the wound is introduced.

Throughout the chapter, the notebooks are treated as the final authority on what was done. Where the research paper that accompanies this project describes something differently, the difference is stated. The four models are referred to by number:

Model	Short name used in this thesis
Model 1	ResNet 50 + Vision Mamba + Hyperedge Attention
Model 2	ConvNeXt B + Vision Mamba + Dynamic Hypergraph + FPN
Model 3	U Net + Vision Mamba + Symbolic Hypergraph
Model 4	ConvNeXt style encoder + Vision Mamba + Multi Granularity Hypergraph

The word "Vision Mamba" is used in this thesis because it is the name used in the notebooks and in the title of the project. Section 3.15 explains precisely what the implemented block does, since it is a simplified relative of the Mamba layer and not a full reproduction of it.

The task solved by all four models is **binary semantic segmentation**: for each input photograph, produce a mask in which pixels belonging to the ulcer region are labelled 1 and all other pixels are labelled 0. The models do not classify an image as healthy or diseased, and they do not diagnose anything. Their output is intended only as a computer assisted aid for delineating a wound region, from which measurements such as area could in principle be derived.

3.2 Overall Processing Pipeline

The processing pipeline, shown in Figure 3.1 and Figure 3.2, has the following steps.

1. Image and mask files are read from five folders belonging to two source datasets (Section 3.3 to 3.6).
2. The images are divided into training, validation and test subsets (Section 3.9).
3. For images that have a mask file, a wound centred crop is taken, the crop is resized to a fixed square size and normalised (Section 3.10). Augmentation is added for training images only.
4. The chosen network produces a logit map, which is compared with the mask through a loss function during training (Sections 3.20 to 3.24).
5. Parameters are updated with AdamW, with warm up and cosine learning rate scheduling, gradient clipping and an exponential moving average of weights (Sections 3.25 to 3.28).
6. After training, predictions on the test subset are produced with six fold test time augmentation and compared with the ground truth using five metrics (Sections 3.30 to 3.32).

Figure 3.1 gives an overview of the complete system and Figure 3.2 the training and inference procedure.

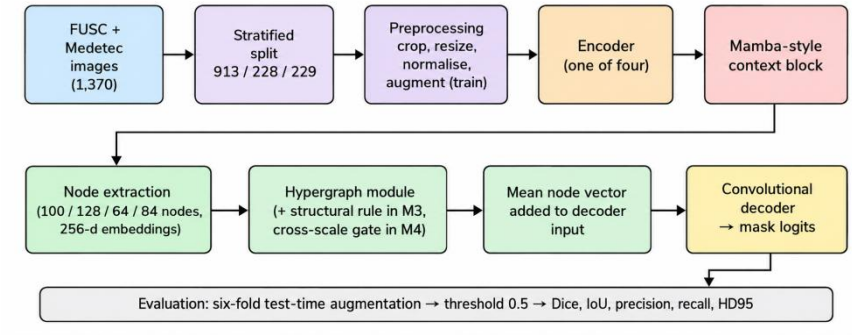


Figure 3.1: Overall system architecture

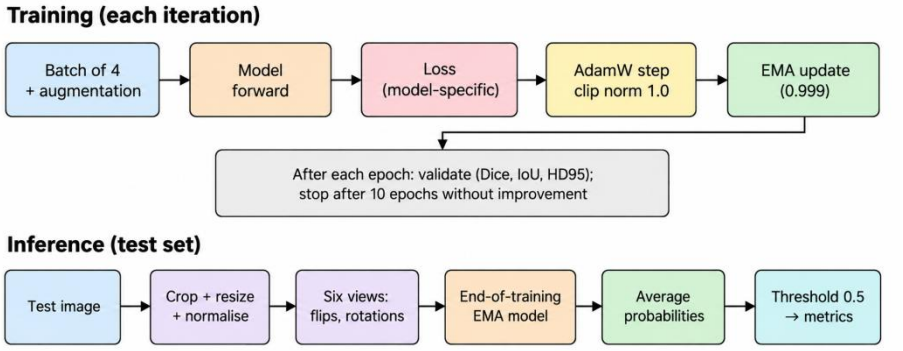


Figure 3.2: Training and inference pipeline

3.3 Dataset Sources

Two public wound image sources were combined. Both were obtained from a single Kaggle dataset collection named “wound seg 3 datasets”, which bundles several public wound datasets. The folder structure found by the notebook is listed in Table 3.1.

3.4 FUSC Dataset

The Foot Ulcer Segmentation Challenge (FUSeg) dataset was created for a challenge held with MICCAI 2021 and described by Wang et al. [18] in *Information*, 15(3), 140. It contains 1,210 foot ulcer photographs: 810 for training, 200 for validation and 200 for testing. The photographs come from 889 patients and were annotated at pixel level by wound care experts. The organisers kept the annotations of the 200 test images private so that submitted methods could be ranked in a blind evaluation, as stated in the challenge description and in several papers that used it (for example, Mahbod et al. [20]).

This detail has a direct consequence for the present work, and it is stated here because it affects how later sections must be read. The notebooks found 810 + 200 labelled images and 200 images in the test folder with no label folder. The unlabelled images are therefore foot ulcer photographs whose reference masks were withheld, not images that are known to contain no wound.

3.5 Medetec Dataset

The Medetec collection used here contains 160 images in total, of which 152 are in a training folder and 8 in a test folder, all with masks. The notebook folder is named `Medetec_foot_ulcer_224`.

Published descriptions of the collection report 152 labelled images showing several types of open foot wounds [20].

3.6 Combined Dataset

The notebooks read all five folders into one dataset object of 1,370 images. The composition is given in Table 3.1.

Table 3.1: Dataset composition as read by the notebooks

Source	Folder	Images	Masks available
FUSC	train	810	yes
FUSC	validation	200	yes
FUSC	test	200	**no (withheld by organisers)**
Medetec	train	152	yes
Medetec	test	8	yes
Total		**1,370**	1,170 with masks

The original train/validation/test boundaries of the two sources were not kept. All 1,370 images were pooled and re split by the procedure in Section 3.9.

3.7 Dataset Structure

Each source follows an `images/` and `labels/` convention, with a mask file sharing the stem of its image file. The loader accepts `.jpg`, `.jpeg`, `.png` and `.bmp` files. A mask file is read in greyscale and converted to binary by thresholding at 127. If no mask file is found for an image, a zero mask of the same size is created and a flag records that no label was loaded. This rule is the origin of the "empty mask" images discussed in Sections 3.11, 3.38 and 4.20.

3.8 Dataset Exploration

Each notebook includes an exploration step that displays sample images, their masks and overlays (Figure 4.2 in Chapter 4). The images vary strongly in wound size, shape, colour, lighting and viewpoint, which is the main reason a single fixed model structure is hard to design by hand for this problem.

3.9 Data Splitting Strategy

The pooled dataset was divided with scikit learn's `StratifiedKFold`, random seed 42, in two stages. The first stage used 3 folds and took the first fold's complement as the training set and its held out part as a temporary set. The second stage split the temporary set into 2 folds, giving the validation and test subsets. Stratification was done on one binary variable only: whether the image's source folder contains labels.

The code comments and the paper describe this as a 70:15:15 split. The arithmetic of the implementation produces something slightly different. Dividing into three folds leaves two thirds of the data for training and one third for the temporary set, and the second stage halves that. The resulting subsets are given in Table 3.2.

Table 3.2: Dataset split produced by the implementation

Subset	Images	Share of 1,370	Images from the unlabelled FUSC folder
Training	913	66.6%	133
Validation	228	16.6%	33
Test	229	16.7%	34

Discrepancy with the project paper. The paper states 70:15:15. The counts 913/228/229 are those produced by the code and are the ones used in all experiments. The per subset unlabelled counts in the last column were obtained by re running the split logic with the same seed and folder sizes, not read from a notebook output.

Figure 3.3 summarises the two stage split.

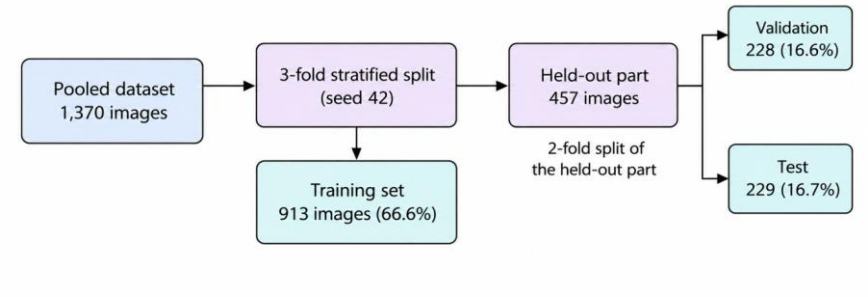


Figure 3.3: Two stage stratified split of the pooled dataset

Because stratification used only the label folder flag, wound size, wound type, patient and source dataset were not balanced across subsets by design. Section 3.38 discusses the consequences.

3.10 Preprocessing

The preprocessing steps are summarised in Table 3.3 and described below.

Table 3.3: Preprocessing operations (all four models)

Step	Operation	Applied to
1	Read RGB image; read mask and binarise at 127	all images
2	Crop to the mask's bounding box plus 10 pixels	images with a mask file
3	Resize to a fixed square (bilinear for images, nearest for masks)	all images
4	Normalise with ImageNet mean and standard deviation	image only

3.10.1 Image Loading

Images are read with PIL and converted to three channel RGB arrays.

3.10.2 Mask Loading

Masks are read in greyscale and set to 1 where the value exceeds 127. For images with no mask file, the mask is all zeros.

3.10.3 Foreground Cropping

When a mask file exists, the image and mask are cropped to the bounding box of the non zero mask region, enlarged by 10 pixels on each side and limited by the image border. If the mask is empty, no crop is made.

This step needs to be understood carefully. The crop box is computed from the **ground truth mask**, and the notebooks apply it in the same way to training, validation and test images. The model is therefore given a wound centred view of every labelled image, including at test time. The consequence is that the reported scores measure how well each network segments a wound **inside a region already known to contain it**. They do not measure the harder problem of finding a wound in a full photograph and then segmenting it. Section 4.36 returns to this point, and it must be kept in mind when the results are compared with studies that segment uncropped images.

Discrepancy with the project paper. The paper mentions foreground cropping without saying it uses the ground truth mask or that it is applied at test time.

Images without a mask file (the 200 withheld label FUSC test images) are not cropped.

3.10.4 Image Resizing

After cropping, each image is resized to a square without preserving the aspect ratio: 512 × 512 for Models 1, 2 and 4, and 384 × 384 for Model 3. Bilinear interpolation is used for images and nearest neighbour for masks, so mask values stay binary.

3.10.5 Image Normalisation

The channel means (0.485, 0.456, 0.406) and standard deviations (0.229, 0.224, 0.225) of ImageNet are used for all models, including the ones whose encoders are not pretrained. These constants are fixed and not computed from the wound data, so no statistic is fitted on the validation or test images.

3.10.6 Training Augmentation

Augmentation is applied only to training images that have a mask file, using a separate dataset object for training. Validation and test use an object with augmentation switched off. The transformations are listed in Table 3.4.

Table 3.4: Training augmentation operations

Operation	Probability / range
Horizontal flip	0.5
Vertical flip	0.5
Rotation by 0, 90, 180 or 270 degrees	one of four chosen uniformly
Elastic deformation (smoothed random displacement field, grid scale 20)	0.3
Brightness, contrast, saturation jitter (each factor 0.8 to 1.2)	0.5

3.10.7 Mask Preserving Augmentation

Flips, rotations and the elastic deformation are applied to the image and mask together. For the elastic deformation the image is remapped with linear interpolation and the mask with nearest neighbour interpolation. Colour jitter is applied to the image only. Figure 3.4 shows the workflow.

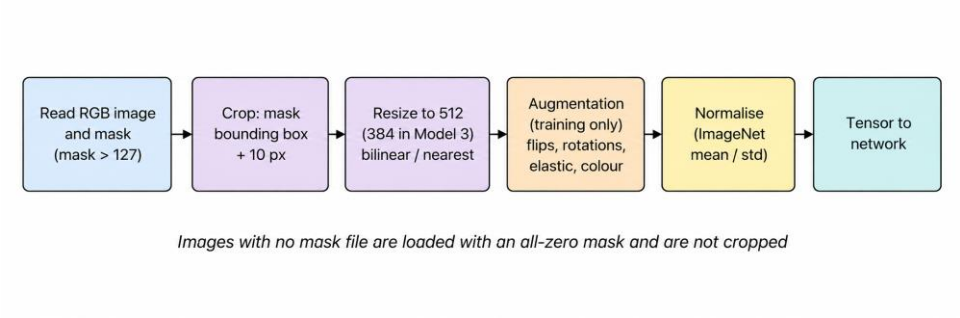


Figure 3.4: Preprocessing workflow

3.11 Data Integrity and Label Handling

Two kinds of image with an all zero mask appear in the data.

1. **Images from the FUSC test folder** (200 images) have no label file. Their masks are zero because the loader supplies a zero mask when none is found. They are real wound photographs whose reference masks are unavailable. About 133 of these fall in the training subset, 33 in validation and 34 in test.
2. **Images with a mask file whose content is empty.** The test subset contains 61 images with an all zero mask in total, and 34 of them come from the unlabelled folder, so 27 come from labelled folders. Where these 27 come from, and whether the masks are truly empty or are emptied by the 127 threshold, was not established from the notebook outputs.

The thesis therefore calls these images **empty mask images**, and does not call them wound free images. Section 3.31 explains how they enter training, validation and testing, and Section 4.20 reports the evaluation on them separately.

During training, empty mask images are used like any other image, with no augmentation. The network is trained to output an empty mask for them. For the unlabelled FUSC images this is a labelling error in the training signal, since the images do contain wounds. The experiments were kept as they were originally run, and this issue is reported as a limitation (Sections 3.38, 4.35 and 4.36).

3.12 Model Development Framework

All code is written in Python with PyTorch (version 2.10.0 with CUDA 12.8 as printed by the notebooks) and runs in Kaggle notebooks. Torchvision supplies ResNet 50 and ConvNeXt B. Random seeds for Python, NumPy and PyTorch are set to 42 and cuDNN is set to deterministic mode with benchmarking off. Each model is a separate notebook and reuses the same data loading and evaluation code with small changes.

3.13 Common Architecture

Figure 3.5 shows the structure shared by the models. For a batch of images the encoder produces feature maps at several scales. The coarsest map goes through a Vision Mamba block, then a fixed number of nodes is taken from the feature maps and given 256 dimensional embeddings. A hypergraph module exchanges information among nodes. Its output is reduced to a single vector per image (the mean over nodes, after a linear projection) that is added to the decoder's input features. A convolutional decoder with skip connections produces the final logit map at the input size.

A property of this design is important for interpreting the later experiments. In every model the decoder receives the hypergraph output only as **one vector per image**, broadcast over all spatial positions, and not as a spatial map. The hypergraph therefore acts as a global context signal, and the

spatial arrangement of nodes is not passed on to the decoder directly. (In Model 4 three vectors are formed, one for each scale.) No experiment in this project removes the hypergraph path, so how much it contributes to the scores is not measured.

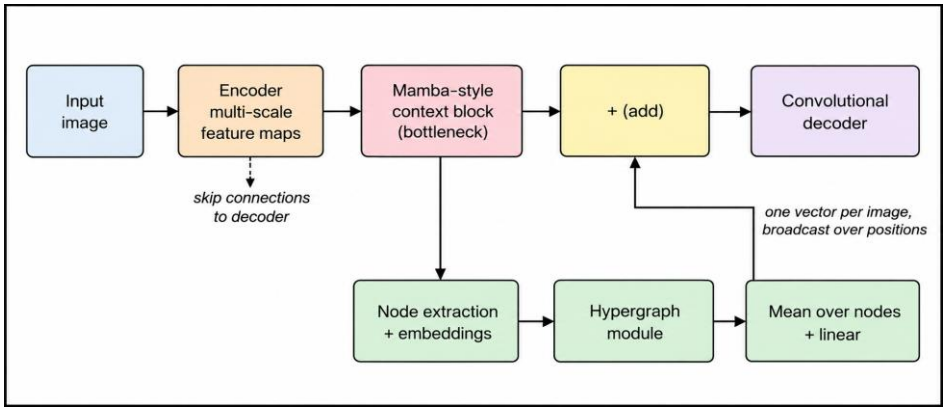


Figure 3.5: Common architecture of the four models

3.14 Vision Mamba Bottleneck

In all four models a block called `SSMBlock` is wrapped in a bottleneck that normalises the feature map with group normalisation, flattens it into a sequence of tokens (one per spatial position), passes the sequence through the block, restores the spatial shape, applies a 1×1 convolution and adds the input back as a residual connection. Table 3.5 lists where it is used.

Table 3.5: Placement of the Vision Mamba block

Model	Where the Vision Mamba block acts	Channels
1	Final ResNet 50 stage (stride 32)	2048
2	Final ConvNeXt B stage (stride 32)	1024
3	U Net bottleneck	512
4	After each of three encoder stages and after the last stage	192, 384, 768, 768

Figure 3.6 shows the bottleneck and the layers inside the block.

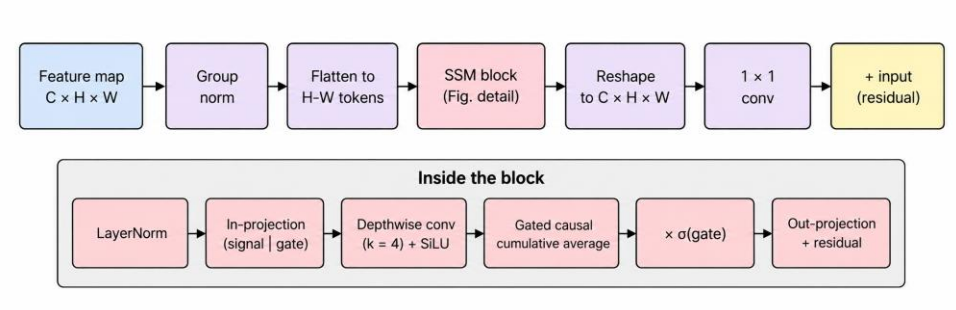


Figure 3.6: Vision Mamba bottleneck and the block it contains

3.15 Selective State Space Model (as implemented)

State space models describe a sequence through a hidden state that is updated at each step. The Mamba model of Gu and Dao [9] makes the update parameters depend on the input and evaluates the recurrence with a hardware aware parallel scan, which keeps the cost linear in sequence length.

The block implemented here keeps the outer structure of a Mamba layer: layer normalisation, an input projection that splits into a signal branch and a gate branch, a short depthwise one dimensional

convolution with kernel 4, a SiLU activation, an input dependent step size, a gated output and a residual connection. The expansion factor is 2 and the nominal state size is 16.

The sequence mixing step, however, is a simplification. Writing x for the signal after the convolution and activation, and Δ for the step size produced by a linear layer from a clamped input, the notebooks compute

where L is the sequence length, D is a learned per channel scale, σ is the sigmoid, and the sum is a cumulative sum along the token sequence. The output is y multiplied by the sigmoid of the gate branch, projected back to the model width, and added to the input. The code defines the state matrix `A_log`, and the projection that would give the input dependent state parameters, but neither is used in Equation (3.1). The block is therefore a gated causal running average along the flattened token order. It does give each token access to all earlier tokens at linear cost, which was the reason for using it, but it is not a state space recurrence.

Equations (1) to (3) of the paper are a further simplified description. *Discrepancy with the project paper.* The paper writes the sum over all positions without the causal restriction and omits the convolution and the gate; the notebook is the reference. For this reason the thesis refers to the component as a "Mamba inspired gated cumulative context block", while continuing to use the term "Vision Mamba bottleneck" for its role in the pipeline. The tokens are ordered row by row, so the context accumulated for a token depends on this fixed raster order; no bidirectional or four direction scan is used.

3.16 Hypergraph Representation

In a graph an edge joins two nodes. In a hypergraph a hyperedge may join any number of nodes, so a group of regions that share a property can be linked in one relation. All models store the structure as an $N \times N$ binary matrix in which entry (i, j) is 1 if node j is in a hyperedge that node i takes part in. Self connections are added, and any node left without a neighbour is forced to connect to itself so that normalisation and attention never see an empty row. Three practical features of the implementation should be noted:

- Adjacency built from feature similarity is computed from the **mean of the node features over the batch**, so one similarity structure is shared by all images in a batch. With the batch size of 4 used in training this mixes images, while at test time the batch size is 1 and the structure comes from the single image. This train/test difference is not examined further in the project.
- In Models 2 and 4 the message passing layers also average node features over the batch before aggregating. The same remark applies.
- The adjacency matrices are not part of the learned parameters. Only the node features and the weights that process them are learned.

3.17 Hypergraph Node Construction

Nodes are taken from feature maps in the following ways.

- **Model 1:** the stride 32 feature map after the Vision Mamba block ($16 \times 16 = 256$ positions at 512×512 input) is flattened and 100 positions are chosen at evenly spaced indices of the flattened sequence. Each is projected linearly from 2048 to 256 dimensions and a learned positional embedding is added.
- **Model 2:** 128 nodes of three kinds. Intensity nodes use patch statistics (mean and standard deviation per colour channel, plus their channel averages) of the input image; texture nodes use a learned 3×3 depthwise convolution applied to the first ConvNeXt stage (named "Gabor

style" in the code, although it is an ordinary learnable filter with one orientation); position nodes use a normalised coordinate grid and a learned embedding. The three sets are concatenated and reduced to 256 dimensions by a linear layer and ReLU.

- **Model 3:** 64 nodes chosen by evenly spaced indices from the 512 channel bottleneck map (48 × 48 positions at 384 × 384 input), projected to 256 dimensions with a learned positional embedding. *Discrepancy with the project paper.* The project description gives an 8 × 8 patch division; the code selects 64 positions by evenly spaced flattened index, which does not form a regular 8 × 8 grid.
- **Model 4:** three sets of nodes obtained by adaptive average pooling of three encoder feature maps: 64 fine nodes (8 × 8 grid, from the stem output), 16 mid nodes (4 × 4 grid, from the first stage) and 4 coarse nodes (2 × 2 grid, from the second stage), each projected to 256 dimensions with positional embedding and layer normalisation. The total is 84 nodes. (The notebook's printed summary labels the grids as 4 × 4, 8 × 8 and 16 × 16; these are patch size labels and do not match the pooling grids, which are those given here..)

Figure 3.7 summarises how nodes, adjacency and message passing fit together.

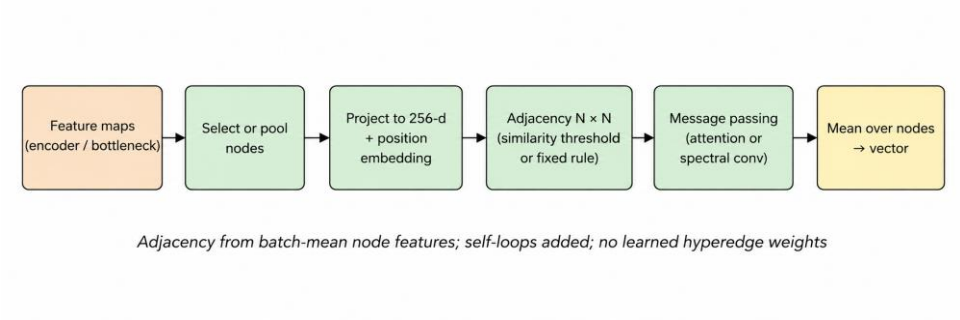


Figure 3.7: Hypergraph construction

3.18 Anatomical Knowledge Representation

Wound care describes a wound in terms of regions with a typical arrangement: the **wound bed** at the centre, the **wound edge** around it, the **perilesional skin** just outside, and **healthy skin** beyond. Models 1 and 3 encode this arrangement by assigning each node an anatomical label based on its **position in the node list**:

Table 3.6: Node index ranges used as anatomical groups

Model	Wound bed	Wound edge	Perilesional	Healthy
1 (100 nodes)	0 to 49	50 to 74	75 to 89	90 to 99
3 (64 nodes)	0 to 31	32 to 47	48 to 55	56 to 63

Table 3.6 gives the ranges. These labels are fixed by index. They are not computed from the image, from the mask or from any segmentation. A node that is labelled "wound bed" is simply one whose index is in the first range, and its image location is wherever the evenly spaced sampling placed it. For the labels to carry anatomical meaning the network would have to learn, indirectly, to make those nodes behave like the corresponding regions, and nothing in the training procedure checks this. The symbolic part of Models 1 and 3 is therefore a structural prior on how information flows among node groups, and not a verified anatomical segmentation. This limitation is considered again in Sections 4.25, 4.26 and 4.35.

Model 2 contains no anatomical rule. Model 4 contains a rule like module of a different kind, described in Section 3.23.

3.19 Neurosymbolic Fusion

"Neurosymbolic" here means that a hand written structural rule, not only learned weights, shapes the computation. The rule used in Models 1 and 3 is that wound edge nodes should be related to wound bed nodes. In Model 1 it is applied by a gating layer: for the edge nodes X_{edge} and the mean embedding $\bar{X}_{\text{bed}}$ of the bed nodes,

so the gate g, computed per node and per feature, decides how far each edge node moves toward the bed mean. Models 1 and 3 each apply this rule once to the nodes. The same rule appears at the node embedding stage in Model 3 and as a symbolic rule layer in Model 1.

3.20 Model 1: ResNet 50 + Vision Mamba + Hyperedge Attention

Encoder. A ResNet 50 initialised with ImageNet weights (`IMAGENET1K_V1`; downloaded at run time) supplies feature maps at strides 4, 8, 16 and 32 with 256, 512, 1024 and 2048 channels. The input is 512 × 512.

Context block. The 2048 channel stage 4 map passes through the Vision Mamba bottleneck of Section 3.14.

Nodes. One hundred nodes are sampled as in Section 3.17 and given anatomical index ranges as in Section 3.18.

Hyperedge structure. Five adjacency matrices are used: 1. *Region*: nodes in the same anatomical group are connected. 2. *Boundary*: edge nodes are connected to bed nodes and bed nodes to edge nodes. 3. *Spatial*: each node is connected to nodes within five positions in the node list. 4. *Colour similarity*: node pairs whose cosine similarity exceeds 0.7, using batch mean features. 5. *Texture similarity*: node pairs whose L1 distance, divided by the largest distance, is below 0.3.

The first three are fixed once; the last two are rebuilt in every forward pass. The "spatial" relation is defined on node list index, not on image coordinates, and it is only approximately spatial because the nodes were sampled at evenly spaced indices from a raster ordered map.

Hyperedge attention. A stack of three attention layers (four heads, width 256, feed forward expansion factor 2, dropout 0.1) is applied to the nodes once for each of the five adjacency matrices, using the matrix as an attention mask so that a node attends only to nodes it is related to. The same three layers are used for all five matrices, so the weights are shared across relation types. The five outputs are concatenated and projected back to 256 dimensions.

Symbolic rule layer. The gate of Equation (3.2) is applied to the input nodes, producing a symbolic version of the nodes.

Neurosymbolic fusion. A cross attention step treats the hyperedge attention output as the query and the symbolic nodes as keys and values, and a learned sigmoid gate mixes the original and cross attended nodes. The result is projected, averaged over nodes and added to a 1 × 1 projected version of the bottleneck feature map.

Decoder. A U Net style decoder with five up sampling blocks (transposed convolution, concatenation with stage 3, stage 2 and stage 1 skip features for the first three blocks, then two blocks without skips) produces a one channel logit map, which is bilinearly resized to 512 × 512.

Loss. $0.4 \times$ binary cross entropy $+ 0.6 \times$ Dice loss (Section 3.24).

Figure 3.8 shows the structure of Model 1 and Figure 3.9 the five hyperedge types drawn on a wound image from the notebook output.

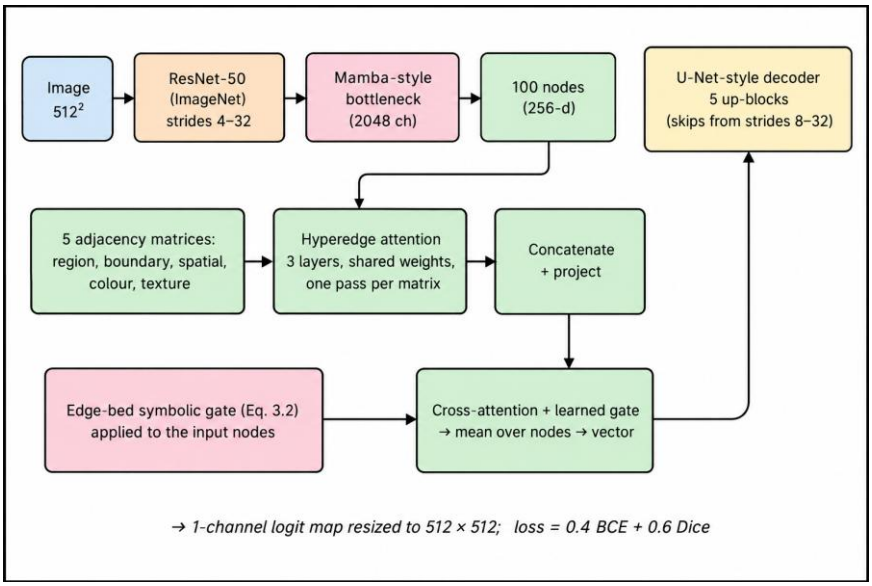


Figure 3.8: Model 1 architecture

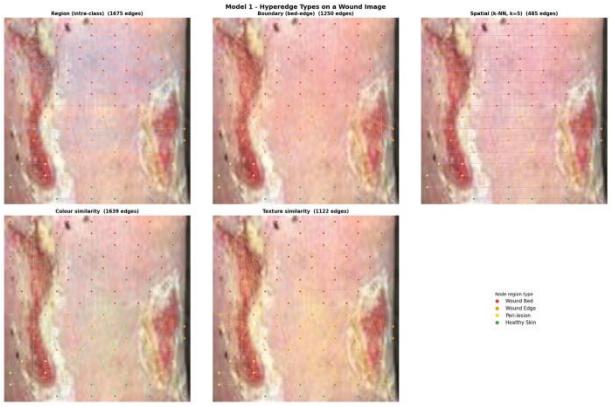


Figure 3.9: Model 1 hyperedge types on a wound image (notebook output)

3.21 Model 2: ConvNeXt B + Vision Mamba + Dynamic Hypergraph

Encoder. ConvNeXt B from torchvision with ImageNet weights (`IMAGENET1K_V1`), split into four stages with strides 4, 8, 16, 32 and 128, 256, 512, 1024 channels. Input 512 × 512.

Context block. Vision Mamba bottleneck on the 1024 channel final map.

Nodes. One hundred and twenty eight nodes combining intensity, texture and position information (Section 3.17), each in 256 dimensions.

Dynamic hypergraph. In every forward pass the node features are L2 normalised and averaged over the batch, and the pairwise cosine similarity is thresholded at 0.6 to form the binary matrix of Equation (3.3). The diagonal is set to 1.

This is “dynamic” in the sense that the connections depend on the current features, as opposed to being fixed in advance. There is no learned hyperedge weight; all weights are 1.

Spectral hypergraph convolution. Three layers of the form $X' = D_v^{-1} H D_e^{-1} H^{\top} X W$ are used, where D_v and D_e are the node and hyperedge degrees, followed by residual connections, layer normalisation and a feed forward block. Following Feng et al. [15] for the general form of hypergraph convolution.

Decoder. A Feature Pyramid Network: 1 × 1 lateral convolutions bring the four encoder maps to 256 channels, a top down pathway adds up sampled coarser maps to finer ones, 3 × 3 convolutions refine each level, a projected mean of the node features is added to the finest level, and a small

convolutional head produces the logits at 512 × 512. This is the only model whose decoder does not use concatenation skip connections in a U Net manner; the FPN adds maps instead.

Loss. 0.4 × focal loss +0.6 × Dice loss, with focal parameters $\gamma = 2$ and $\alpha = 0.25$.

Symbolic reasoning. None. The paper's Table 1 agrees with the code on this point.

Figure 3.10 shows the structure of Model 2 and Figure 3.11 the node fusion and dynamic hypergraph steps as drawn by the notebook.

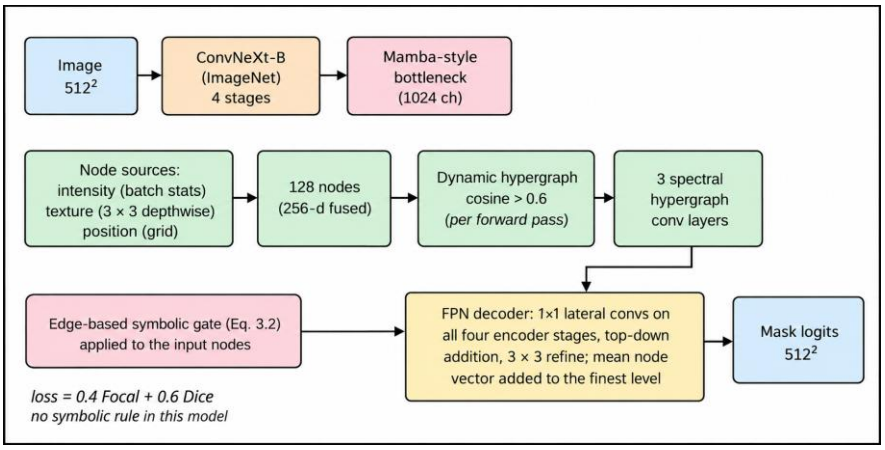


Figure 3.10: Model 2 architecture

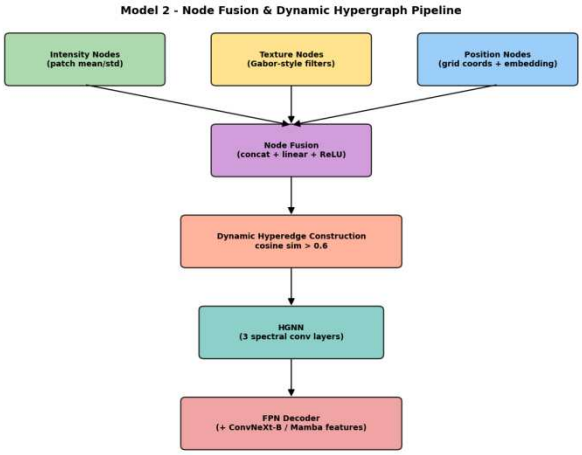


Figure 3.11: Model 2 node fusion and dynamic hypergraph pipeline (notebook output)

3.22 Model 3: U Net + Vision Mamba + Symbolic Hypergraph

Encoder. A three stage U Net encoder built from convolution and residual blocks (64, 128, 256 channels, max pool down sampling), followed by a 512 channel bottleneck. **It has no pretrained weights.** Input 384 × 384.

Context block. Vision Mamba bottleneck on the 512 channel map.

Nodes. Sixty four nodes with anatomical index ranges (Table in Section 3.18) and the edge to bed gating of Equation (3.2) already applied inside the node embedding.

Two hypergraphs. A *visual* adjacency from colour and texture similarity (cosine above 0.7 or normalised L1 distance below 0.3, combined by union) and a *symbolic* adjacency joining wound bed to edge, edge to perilesional, perilesional to healthy and nodes within each group. Messages are the adjacency weighted mean of linearly projected node features.

Region wise fusion gate. There are four learnable scalars θ_r, one per anatomical region, each initialised to 0.5. The fused node is

so every region starts with $\alpha_r = \sigma(0.5) \approx 0.622$ for all four regions, and training can move each region's value independently. The values at the end of training, and their relation to this initial value, are analysed in Section 4.26.

Decoder. A U Net decoder with three up sampling blocks and skip connections. The mean node feature is projected and added to the bottleneck before decoding.

Loss. 0.5 × Lovász hinge loss (per image) +0.5 × binary cross entropy.

Figure 3.12 shows the structure of Model 3 and Figure 3.13 the symbolic and visual hyperedges on a sample of nodes.

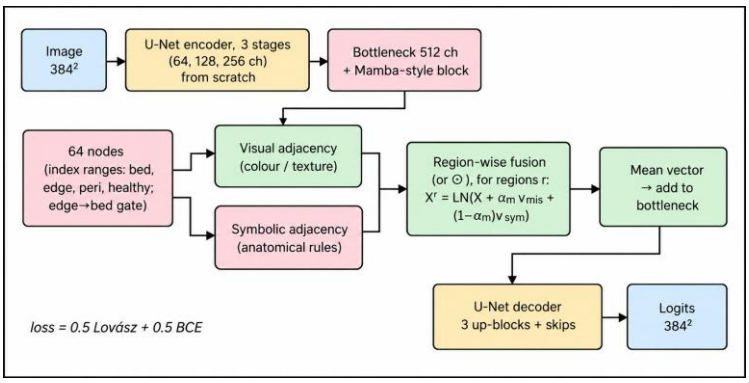


Figure 3.12: Model 3 architecture

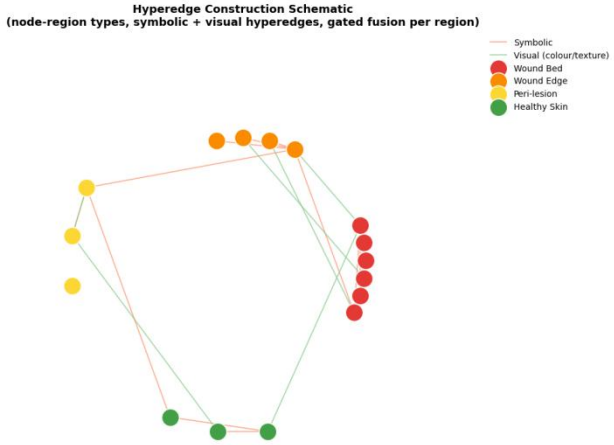


Figure 3.13: Model 3 symbolic and visual hyperedges over the four region types (notebook output)

3.23 Model 4: Multi Granularity Hypergraph

Encoder. The encoder is a custom network, not the torchvision ConvNeXt B. It starts with a ConvNeXt style 4 × 4 stride 4 patchify convolution with normalisation (96 channels), followed by three stages each made of a residual convolution block, a stride 2 down sampling convolution, and a Vision Mamba bottleneck (192, 384, 768 channels), and a final deep Vision Mamba block. **The code imports the ConvNeXt B weight enumeration but does not load pretrained weights, so the encoder is trained from scratch.** *Discrepancy with the project paper.* The model is called "ConvNeXt + Vision Mamba" in the paper and project notes; only the stem is ConvNeXt like. The thesis uses the name "ConvNeXt style". Input 512 × 512.

Nodes at three granularities. Sixty four fine, 16 mid and 4 coarse nodes by adaptive average pooling (Section 3.17), 84 in total.

Hypergraph. Within each level, nodes are connected where cosine similarity exceeds 0.6 (batch mean features, self loops). Across levels, each fine node is joined to the mid node covering the same area, and each mid node to its coarse node, by integer division of grid coordinates. The full 84 × 84 matrix is built from these blocks.

Hierarchical message passing. Three "bottom up" layers and three "top down" layers, with separate weights, are applied in sequence. The top down layers use the transpose of the adjacency matrix. Since the matrix is symmetric by construction, the transpose equals the original, so the two passes differ in their weights and position in the stack but not in the connections they follow.

Scale gating module. A module named `WoundAnatomySymbolicFusion` pulls the fine nodes toward the mean of the mid nodes, the coarse nodes toward the mean of the mid nodes, the mid nodes toward the mean of the coarse nodes, and the fine nodes again toward the new mid mean, each with a learned sigmoid gate. This module encodes a relationship between **scales**. It does not contain the wound bed and wound edge rule of Equation (3.2) or any reference to anatomical regions. *Discrepancy with the project paper.* The paper's Table 1 and Section 3.6 state that Model 4 uses edge bed gating. The notebook's module is a cross scale gate; the thesis follows the code.

Decoder and deep supervision. The node features of each scale are projected, averaged and added to the bottleneck with weights 0.3 (fine), 0.4 (mid) and 0.3 (coarse). A U Net style decoder with four up sampling blocks follows. During training, three auxiliary 1 × 1 heads on the outputs of the first three decoder blocks give extra masks at the input size.

Loss. For each prediction, 0.5 × BCE +0.5 × boundary weighted Dice loss. The total is the main loss plus the auxiliary losses weighted 0.4, 0.3 and 0.2 (Section 3.24.6).

Figure 3.14 shows the structure of Model 4, Figure 3.15 its three level node pyramid and Figure 3.16 the order of the message passing layers.

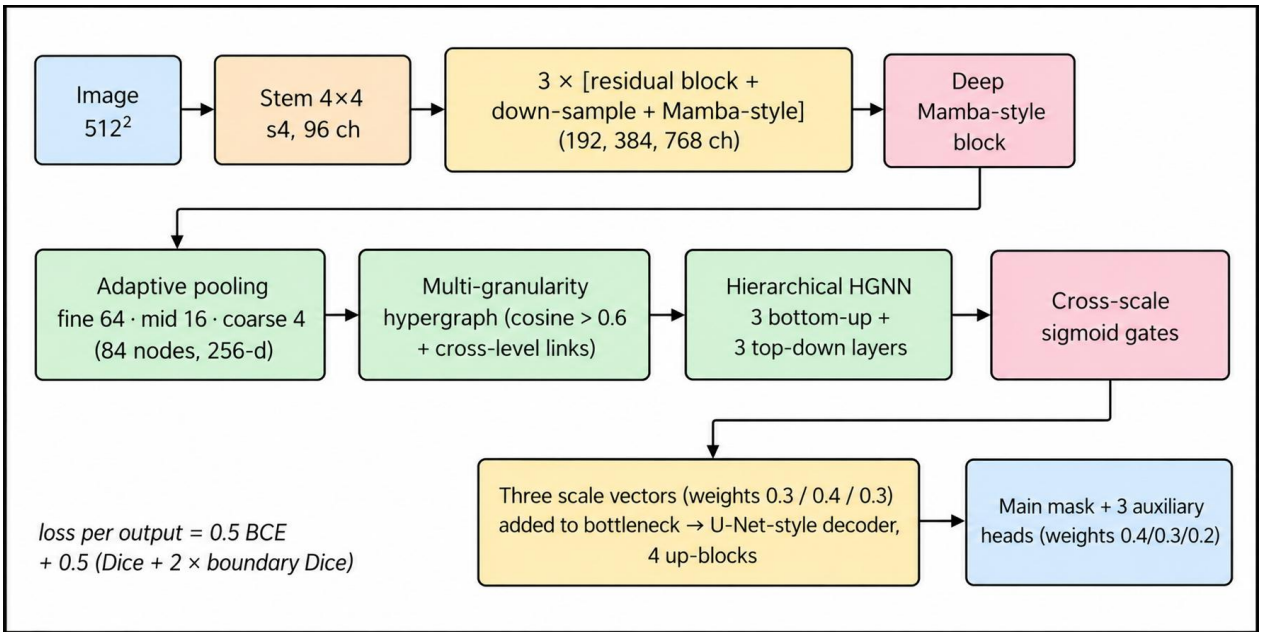


Figure 3.14: Model 4 architecture

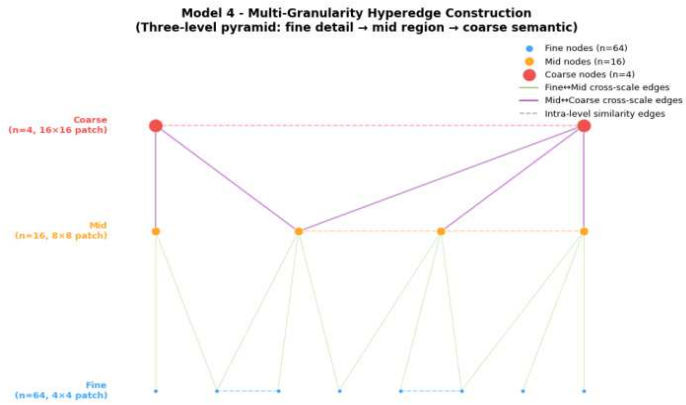


Figure 3.15: Model 4 multi granularity hyperedge construction (notebook output)

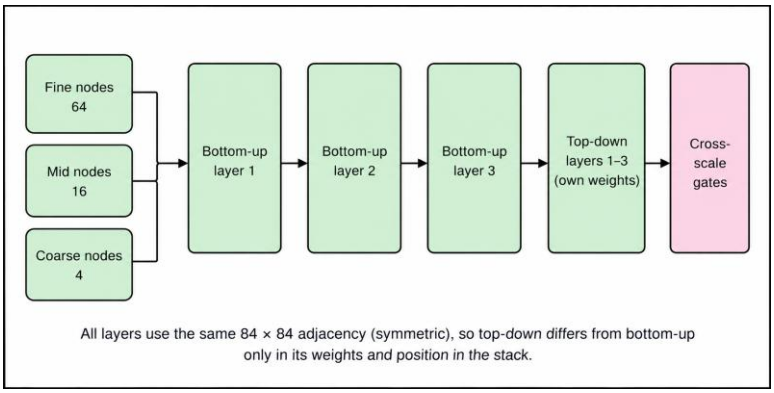


Figure 3.16: Hierarchical message passing in Model 4

Table 3.7: Model architecture comparison

Component	**Model 1**	**Model 2**	**Model 3**	**Model 4**
Encoder	ResNet 50 (ImageNet)	ConvNeXt B (ImageNet)	3 stage U Net (scratch)	ConvNeXt style stem + residual/Mamba stages (scratch)
Input size	512	512	384	512
Nodes	100	128	64	64 + 16 + 4 = 84
Hyperedges	five relation types (3 fixed, 2 similarity)	one similarity based, rebuilt each pass	visual + symbolic	within level similarity + cross level links
Reasoning layers	3 attention layers per relation, shared	3 spectral HGNN layers	one fusion step	3 bottom up + 3 top down layers
Anatomical rule	edge bed gating	none	edge bed gating + symbolic adjacency	cross scale gating only
Decoder	U Net	FPN	U Net	U Net style with deep supervision

Table 3.8: Model specific loss functions

Model	**Loss**
1	0.4 BCE + 0.6 Dice
2	0.4 Focal ($\gamma = 2$, $\alpha = 0.25$) + 0.6 Dice
3	0.5 Lovász hinge + 0.5 BCE
4	0.5 BCE + 0.5 (Dice + 2 × boundary Dice), plus 0.4/0.3/0.2 auxiliary terms

Table 3.9: Model specific input and node configuration

Model	Input	Nodes	Node dimension
1	512 × 512	100	256
2	512 × 512	128	256
3	384 × 384	64	256
4	512 × 512	84 (64 + 16 + 4)	256

The four models differ at once in encoder, pretraining, input size, number of nodes, loss and decoder. A difference in scores between two models therefore cannot be traced to any single design choice. Chapter 4 treats the comparison as a comparison of whole designs, not as a test of one component.

3.24 Loss Functions

In what follows $p_i \in [0,1]$ is the predicted probability and $g_i \in \{0,1\}$ the reference label for pixel i.

3.24.1 Dice Loss

computed per image with $\epsilon = 1$ and averaged over the batch. It measures overlap directly and is less affected than cross entropy by the large number of background pixels.

3.24.2 Binary Cross Entropy

It gives every pixel an independent penalty and provides stable gradients.

3.24.3 Focal Loss

where p_t is the probability assigned to the true class. The factor $(1 - p_t)^\gamma$ reduces the weight of pixels that are already classified confidently. In the code a single constant $\alpha = 0.25$ multiplies all pixels and no separate weighting is given to the two classes.

3.24.4 Lovász Loss

The Lovász hinge is a convex surrogate for the Jaccard (IoU) loss computed from sorted pixel errors [31]. It is evaluated per image and combined with BCE in Model 3.

3.24.5 Boundary Dice Loss

A boundary band is formed by marking pixels where a 3 × 3 average of the mask is strictly between 0.01 and 0.99, which selects mask pixels near the edge. A Dice loss is computed on this band only and added with weight 2 to the ordinary Dice loss:

When the mask has no boundary pixels anywhere in the batch, only the ordinary Dice term is used.

3.24.6 Deep Supervision Loss

With the main output $\hat{y}_0$ and three auxiliary outputs $\hat{y}_1, \hat{y}_2, \hat{y}_3$,

Auxiliary heads are active only during training.

3.25 Optimization

All models use AdamW with initial learning rate 10^{-4} and weight decay 10^{-4}, a batch size of 4 (training batches drop the last incomplete batch), up to 50 epochs, two data loader workers and dropout 0.1 inside the decoders and attention/feed forward layers. A training step whose loss is not finite is skipped.

Table 3.10: Common training configuration

Setting	Value
Optimiser	AdamW, weight decay 10^{-4}
Initial learning rate	1×10^{-4}
Warm up	5 epochs, linear
Schedule after warm up	cosine annealing to 0 over the 50 epoch horizon
Batch size	4
Maximum epochs	50
EMA decay	0.999
Gradient clipping	max norm 1.0
Early stopping	patience 10 epochs on validation Dice IoU mean
Seed	42

3.26 Learning Rate Scheduling

The rate is changed once per epoch. For epoch index e (starting at 0) with $W = 5$ warm up epochs and $T = 50$ total epochs, the multiplier is $(e + 1)/W$ for $e < W$, and $1/2\left[1 + \cos\left(\pi(e - W)/(T - W\right.\right.$ afterwards. Warm up avoids large early updates in randomly initialised layers; the cosine decay brings the rate smoothly close to zero by the end.

3.27 Exponential Moving Average

A shadow copy of all entries of the model's state dictionary (weights and normalisation buffers) is updated after every optimiser step:

The average smooths out step to step noise. **During training, validation is performed with the ordinary weights, not the averaged weights.** The EMA weights are used only in the final test evaluation (Section 3.31).

3.28 Gradient Clipping

After each backward pass the global gradient norm is limited to 1.0. The number of clipped steps per epoch is counted by the code. These counts are not printed in the epoch logs and are not analysed in this thesis.

3.29 Early Stopping

The validation score is $0.5\,\text{Dice} + 0.5\,\text{IoU}$. When it does not exceed its best value for 10 consecutive epochs, training stops. The ordinary weights at each new best score are saved as the "best" checkpoint. Models 1, 3 and 4 completed all 50 epochs without early stopping; Model 2 stopped after epoch 34. (These are facts about the runs, listed in Table 4.2.)

3.30 Test Time Augmentation

At test time each image is processed six times: unchanged, flipped horizontally, flipped vertically, and rotated by 90, 180 and 270 degrees. Each prediction is transformed back to the original orientation and the six probability maps are averaged. A threshold of 0.5 is then applied to give the binary mask. TTA is applied only during inference, never during training, and uses no information about the reference mask.

3.31 Prediction Procedure

The final test evaluation in each notebook proceeds as follows. First the file of the best scoring checkpoint is loaded; then the EMA weights saved at the end of training are applied to the model, which **replaces** the loaded weights. The model evaluated on the test set is therefore the end of training EMA model, and not the best validation checkpoint. The validation scores reported in the logs belong to the ordinary weights and the test scores to the EMA weights, so the two are not scores of the same model. For Model 2, which stopped early, the EMA is that of epoch 34. *Discrepancy with the project paper.* The paper says model selection was based on validation score and does not mention this.

Test images are processed one at a time. Two sets of metrics are produced:

- **Lesion containing test images** (168): images whose reference mask has at least one positive pixel.
- **Full test set** (229): all images, including the 61 empty mask images.

The headline results in this thesis are those on the 168 lesion containing images, as in the paper. The full test set results are reported separately in Section 4.20 with the caveat of Section 3.32. *Discrepancy with the project paper.* The paper states that wound free images were evaluated "separately using classification based measures". No such measures exist in the notebooks.

3.32 Evaluation Metrics

Let TP, FP, FN be true positive, false positive and false negative pixel counts for an image. The notebooks add a constant 1 to numerator and denominator of each overlap metric, which avoids division by zero when a mask is empty. The effect on images with large wounds is negligible; it matters for empty or very small masks.

Table 3.11: Evaluation metrics

Metric	Definition as implemented
Dice	$(2\,TP + 1)/(2\,TP + FP + FN + 1)$
IoU	$(TP + 1)/(TP + FP + FN + 1)$
Precision	$(TP + 1)/(TP + FP + 1)$
Recall	$(TP + 1)/(TP + FN + 1)$
HD95	95th percentile of pooled boundary to boundary distances, in pixels

3.32.1 Dice Coefficient

The overlap measure of Equation (3.5) without the loss sign. It lies between 0 and 1.

3.32.2 Intersection over Union

The ratio of the intersection of predicted and reference regions to their union. For one image IoU = Dice / (2 − Dice) apart from the smoothing constants.

3.32.3 Precision

The share of predicted wound pixels that are correct. Low precision indicates over segmentation.

3.32.4 Recall

The share of reference wound pixels that are found. Low recall indicates under segmentation.

3.32.5 Hausdorff Distance at the 95th Percentile

Boundary pixels of the prediction and of the reference are obtained by subtracting a binary erosion from each mask. For every boundary pixel of one mask the distance to the nearest boundary pixel of the other mask is found, in both directions, and the two sets of distances are pooled; HD95 is their 95th percentile (Hausdorff distance [32]). It is measured in pixels of the **resized network input** (512 × 512 or 384 × 384, taken after cropping), so the units are not millimetres and not comparable across the two input sizes, and the values depend on how large the crop was relative to the wound. Three conventions in the code matter:

- if both masks are empty, HD95 is 0;
- if exactly one mask is empty, the distance is undefined and the image is skipped when HD95 is averaged;
- when a single image has an undefined distance and is therefore skipped, the code records an HD95 of **0.0** for it.

The last rule means that an image on which the model outputs an empty mask for a visible lesion is entered with HD95 = 0, the best possible value, which slightly flatters the mean HD95 of models that fail completely on some images. The number of lesion containing test images with an empty prediction was not counted for each model; the minimum Dice of 0.0 reported for Model 4 shows that at least one such image exists there.

Metrics are computed per image, then averaged over images. Mean, standard deviation, minimum and maximum are reported, and 95% confidence intervals for the mean are obtained by bootstrap resampling of the per image values (10,000 resamples, seed 42, percentile interval).

3.33 Model Complexity

The number of trainable parameters is taken from PyTorch. Floating point operations (FLOPs) for one 512 × 512 input are measured with the `thop` library in Models 1 and 4 (multiply accumulate counts converted to FLOPs by doubling). In Models 2 and 3 the library was not installed in the run, so no FLOP figure exists for them; they are reported as “not measured”. For Model 1 the notebook also gives the parameter count of each module, which is useful because the Vision Mamba block with 2048 channels holds about 29.6 million of 61.7 million parameters, more than the ResNet 50 encoder itself (23.5 million). Parameter counts of all four models are given in Table 4.8.

3.34 Inference Time

Single image latency without TTA was measured in the notebooks on the Kaggle GPU, from which frames per second are derived. Latency with six fold TTA is not measured; it is given as six times the single pass latency, which the printed values confirm. These numbers describe the experimental environment only. They are not a statement about clinical real time performance, and they were not obtained on any device suitable for deployment. The number of timing repetitions, the use of warm up passes and whether the GPU was synchronised before reading the clock were not recorded; a batch size of 1 is assumed.

3.35 Explainability and Architectural Analysis

No post hoc attribution method such as Grad CAM or SHAP was used. The interpretability material consists of examining internal quantities of the models, listed in Table 3.12. These can show how a component behaved on the data, but they do not show that the component caused a given prediction.

Table 3.12: Architectural analyses implemented

Model	Analysis
1	Edge bed symbolic gate values; the five adjacency structures drawn over a sample image; hypergraph schematic
2	Contribution of the three node branches; node degree and adjacency matrix of the dynamic hypergraph
3	Learned values of the four fusion gate parameters; hyperedge adjacency
4	Gate values; activation magnitude of fine, mid and coarse nodes; multi scale hypergraph drawing

3.36 Experimental Environment

Table 3.13: Hardware and software environment

Item	Value
Platform	Kaggle notebooks
GPU	Two NVIDIA T4 GPUs (as stated by the authors); the code selects the single device cuda and uses no multi GPU wrapper
Framework	PyTorch 2.10.0+cu128
Other library versions (torchvision, NumPy, SciPy, scikit learn, OpenCV, thop)	Not recorded
Training time per model	Not recorded
Mixed precision	not used (no autocast in the code)

3.37 Data Leakage Considerations

The following points apply to this work.

- **How the dataset was assembled.** Two public sources were pooled and re split at random in the notebooks. Neither source’s original train/validation/test boundaries were kept.
- **Patient level separation.** The FUSeg dataset contains 1,210 photographs from 889 patients, so some patients have several images. The loader has no patient identifier, and the split is by image. Patient level separation could not be independently verified from the available dataset structure, and images of the same wound or patient may well appear in both training and test subsets. This is likely to inflate scores, to an extent that cannot be measured here.

- **Overlap between sources.** The FUSeg dataset is described as an extension of an earlier chronic wound dataset, and whether any Medetec images also appear in FUSC was not checked.
- **Preprocessing statistics.** Normalisation uses fixed ImageNet constants and nothing is fitted on the data.
- **Augmentation.** Applied to training images only, through a separate dataset object.
- **Test time augmentation.** Used only at inference, and does not use the reference mask.
- **Ground truth cropping.** Test images with masks are cropped using the **reference mask**, so information about the answer enters the test input (Section 3.10.3). This is a stronger issue than the patient overlap question and it is stated plainly in the limitations.
- **Test data in model selection.** The test subset is not used to choose epochs or weights, but the final test model is the end of training EMA model rather than the validation selected checkpoint (Section 3.31).
- **Empty and unlabelled images.** About 133 unlabelled images with all zero masks are in training, 33 in validation and 34 in test (Section 3.11). They shape training and validation and enter the full test set scores.
- **Acquisition characteristics.** The two sources differ in camera, framing and resolution, and the split was not stratified by source, so differences in acquisition may be unevenly distributed across subsets.

CHAPTER 4
EXPERIMENTAL RESULTS AND ANALYSIS

4.1 Experimental Overview

Four networks were trained and tested on the same combined wound image collection, with identical optimiser settings, schedule, random seed and evaluation code (Chapter 3). This chapter reports what happened: how the data were divided, how training behaved, how each network scored on the test images, what the computational cost of each was, and what the internal quantities that were recorded suggest about the components.

Two points govern how every table below should be read.

First, **the headline test scores are computed on the 168 test images that have a reference mask with at least one wound pixel**. Those images were cropped around the reference mask before being given to the network (Section 3.10.3). The scores therefore describe segmentation inside a wound centred region.

Second, **the four models differ in several respects at once** (encoder, whether the encoder was pretrained, input size, node construction, loss, decoder). The experiments compare four complete designs. They are not controlled tests of individual components, and no baseline without the Mamba style block or without the hypergraph was trained. Statements about "the effect of the hypergraph" are therefore not available from these experiments, and this chapter does not make them.

4.2 Dataset Statistics

The notebooks read five folders from two sources, listed in Table 4.1.

Table 4.1: Dataset distribution

Source	Folder	Images	Reference masks
FUSC	train	810	810
FUSC	validation	200	200
FUSC	test	200	none (withheld by the challenge organisers)
Medetec	train	152	152
Medetec	test	8	8
Total		1,370	1,170

Figure 4.1 shows the same counts as a chart. The notebooks do not record wound area statistics, image resolutions or the number of patients per subset.

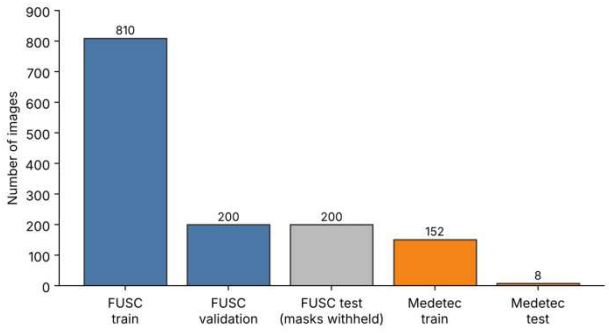


Figure 4.1: Number of images in each source folder

4.3 Dataset Split

The split reproduced in all four notebooks is 913 training, 228 validation and 229 test images, which is 66.6 : 16.6 : 16.7 per cent. The paper's statement of a 70 : 15 : 15 split is not what the code produced (Section 3.9). The test set contains **168 images with a non empty reference mask and 61 images with an all zero mask**. Of the 61, 34 are expected from the unlabelled FUSC test folder (Section 3.11); the origin of the other 27 has not yet been established.

4.4 Sample Image and Mask Analysis

Figure 4.2 shows representative images, reference masks and overlays from the notebook's exploration step. The samples differ widely in wound size, shape, colour, lighting and framing. Some reference masks have visibly blocky, stair stepped edges. The Medetec folder is named "224", which suggests that these images were small originally and were enlarged to the 512 pixel network input, which would explain the blocky masks. This matters for boundary metrics: when an image has been enlarged, a distance of a few pixels in HD95 corresponds to a much smaller distance in the original image.

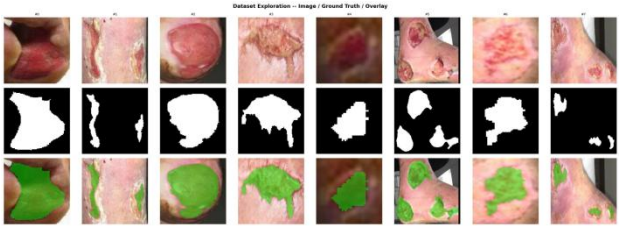


Figure 4.2: Sample images, reference masks and overlays (notebook output)

4.5 Training Behaviour

All networks were trained on a single GPU for up to 50 epochs (228 batches of 4 per epoch), with validation after every epoch. The training time per model was not recorded.

An important property of the validation numbers. The validation score is computed on all 228 validation images, including the empty mask images (at least 33 of them come from the unlabelled folder). An image whose reference and prediction are both empty scores Dice 1 and IoU 1 under the smoothing used in the code. The validation Dice and IoU in Table 4.2 are therefore **not on the same footing** as the test values in Table 4.3, which exclude empty mask images. Validation and test values must not be compared directly.

Table 4.2: Training and validation summary (validation scores computed on all 228 validation images with the ordinary, non EMA weights)

Model	Epochs run	Epoch of best score	Best val Dice	Best val IoU	Best val combined	Early stop
1	50 / 50	48	0.9357	0.8893	0.9125	no
2	34 / 50	24	0.9406	0.8951	0.9178	yes
3	50 / 50	45	0.8811	0.8131	0.8471	no
4	50 / 50	41	0.9111	0.8508	0.8809	no

(The epoch of best score is the last epoch marked as saved in the log.)

Figure 4.3 and Figure 4.4 are the saved training curves.

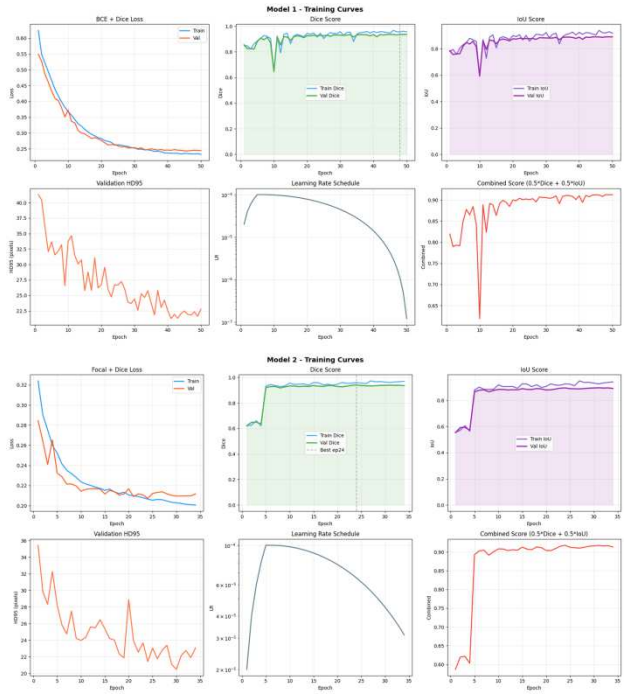


Figure 4.3: Training curves of Model 1 (top) and Model 2 (bottom)

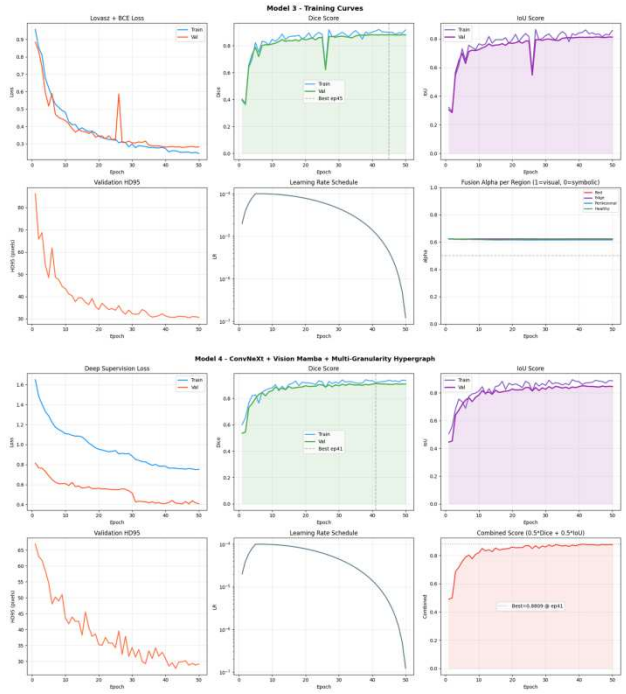


Figure 4.4: Training curves of Model 3 (top) and Model 4 (bottom)

4.6 Model 1 Results

Model 1 (ResNet 50 + Vision Mamba + Hyperedge Attention, 61.73 M parameters) was trained for the full 50 epochs. Validation Dice rose quickly in the first epochs because the encoder was pretrained, then showed one pronounced dip: at epoch 10 the validation Dice fell to 0.647, from 0.880 at epoch 5, before recovering above 0.92 by epoch 20. The logs do not explain the dip; a learning rate peak after warm up, or a batch of unstable updates, are possible but untested explanations. After epoch 30 the validation Dice changed little (0.930 to 0.936).

On the 168 lesion containing test images, with EMA weights and six fold TTA, Model 1 reached Dice 0.9365, IoU 0.8838, precision 0.9412, recall 0.9364 and HD95 21.71 pixels (Table 4.3).

4.7 Model 2 Results

Model 2 (ConvNeXt B + Vision Mamba + Dynamic Hypergraph + FPN, 99.57 M parameters) reached its best validation score at epoch 24. Patience was exhausted at epoch 34 and training stopped. Its validation curve was smooth, with no collapse comparable to Model 1's epoch 10 dip: after epoch 5 its validation Dice never fell below 0.917.

Test results: Dice 0.9367, IoU 0.8843, precision 0.9341, recall 0.9451, HD95 23.43 pixels.

One caution about early stopping. The EMA weights used at test time were those saved at the end of training (epoch 34), ten epochs after the best validation epoch. The reported test score is therefore for the epoch 34 EMA model and not for the epoch 24 checkpoint (Section 3.31).

4.8 Model 3 Results

Model 3 (three stage U Net + Vision Mamba + Symbolic Hypergraph, 13.30 M parameters, 384 × 384 input, not pretrained) learned slowly. Validation Dice was 0.395 after epoch 1 and 0.843 after epoch 20, and it was still rising at epoch 40 (0.880) when the cosine schedule brought the learning rate near zero. Training Dice at the end (0.916) was well above validation Dice (0.880). The slow rise suggests that the schedule, not convergence, ended the training, although this was not tested by longer training.

Test results: Dice 0.8759, IoU 0.7960, precision 0.8748, recall 0.9001, HD95 35.21 pixels. The spread over images was the widest of the four models for Dice (standard deviation 0.123, minimum 0.123).

4.9 Model 4 Results

Model 4 (ConvNeXt style encoder + Vision Mamba + Multi Granularity Hypergraph, 35.75 M parameters, not pretrained) trained for 50 epochs with deep supervision. The reported training loss is the sum of four weighted terms and is not comparable in value with the losses of the other models. Validation Dice reached 0.911 at epoch 41 and was 0.908 at epoch 50.

Test results: Dice 0.9007, IoU 0.8305, precision 0.8558, recall 0.9703, HD95 28.69 pixels. The minimum per image Dice, IoU and recall on the test set are all 0.0000, and the minimum HD95 is 0.0. These values are what the code produces for an image on which the model predicted no wound pixel at all although the reference mask contains wound pixels; at least one such image exists among the 168. The number of such images was not counted for each model. For such an image HD95 is recorded as 0, the best possible value, so the mean HD95 of 28.69 pixels is slightly flattered (Section 3.32.5).

Table 4.3: Test set performance on the 168 lesion containing images (EMA weights, six fold TTA; mean, standard deviation, minimum, maximum over images)

Model	Dice	IoU	Precision	Recall	HD95 (px)
1	0.9365 (0.0456; 0.7459 to 0.9885)	0.8838 (0.0751; 0.5948 to 0.9773)	0.9412 (0.0557; 0.7024 to 1.0000)	0.9364 (0.0657; 0.6147 to 0.9974)	21.71 (21.53; 4.00 to 154.64)
2	0.9367 (0.0461; 0.7612 to 0.9894)	0.8843 (0.0763; 0.6145 to 0.9789)	0.9341 (0.0672; 0.6710 to 1.0000)	0.9451 (0.0609; 0.6166 to 0.9987)	23.43 (28.79; 4.12 to 268.43)
3	0.8759 (0.1233; 0.1229 to 0.9875)	0.7960 (0.1569; 0.0655 to 0.9753)	0.8748 (0.1536; 0.1422 to 0.9993)	0.9001 (0.1263; 0.1082 to 1.0000)	35.21 (33.01; 3.61 to 303.71)
4	0.9007 (0.1027; 0.0000 to 0.9819)	0.8305 (0.1225; 0.0000 to 0.9645)	0.8558 (0.1087; 0.1974 to 1.0000)	0.9703 (0.0851; 0.0000 to 1.0000)	28.69 (22.44; 0.00 to 146.37)

4.10 Comparative Model Performance

Table 4.4 ranks the models on each metric.

Table 4.4: Model comparison (rank 1 is best on that metric)

Metric	Rank 1	Rank 2	Rank 3	Rank 4
Dice	Model 2 (0.9367)	Model 1 (0.9365)	Model 4 (0.9007)	Model 3 (0.8759)
IoU	Model 2 (0.8843)	Model 1 (0.8838)	Model 4 (0.8305)	Model 3 (0.7960)
Precision	Model 1 (0.9412)	Model 2 (0.9341)	Model 3 (0.8748)	Model 4 (0.8558)
Recall	Model 4 (0.9703)	Model 2 (0.9451)	Model 1 (0.9364)	Model 3 (0.9001)
HD95 (lower is better)	Model 1 (21.71)	Model 2 (23.43)	Model 4 (28.69)	Model 3 (35.21)

Model 2 obtained the highest Dice and IoU of the four evaluated architectures. Model 1 obtained the highest precision and the lowest HD95. Model 4 obtained the highest recall. No model leads on every metric, and the first two models are very close to each other. The statement that one network is the best overall is not supported, and none of these results is evidence about suitability for clinical use.

The models fall into two groups. Models 1 and 2 score about 0.937 Dice and Models 3 and 4 score 0.876 and 0.901. The two higher scoring models are the two with ImageNet pretrained encoders, and the two lower scoring ones are trained from scratch. This pattern is consistent with pretraining being an important factor, but the models also differ in input size, loss, node construction and decoder, so the experiments cannot separate pretraining from these other differences.

Figure 4.5 shows the four mean test scores of each model with their 95% bootstrap intervals.

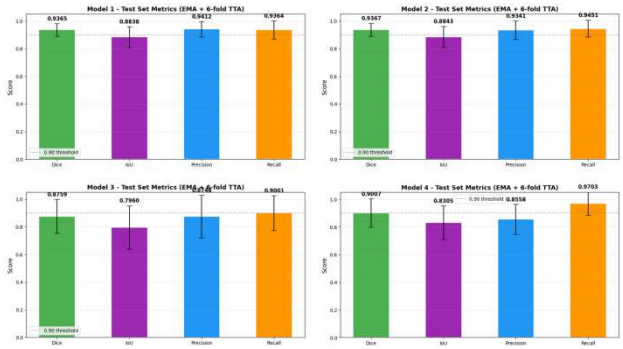


Figure 4.5: Test set metrics (EMA weights, six fold TTA) of Models 1 to 4 with 95% bootstrap intervals

4.11 Dice Analysis

Models 1 and 2 differ by 0.0002 in mean Dice. Their 95% bootstrap intervals, [0.9295, 0.9432] and [0.9296, 0.9433], are almost identical, and each interval is about 0.014 wide, which is roughly seventy times the difference between the means. The two are for practical purposes tied on Dice. Model 4 is about 0.036 below them and its interval [0.8833, 0.9146] does not overlap theirs. Model 3's interval [0.8559, 0.8931] overlaps Model 4's slightly (0.8833 to 0.8931). Overlap of intervals is not a statistical test, and no paired test on per image scores was done.

The spread of per image Dice is much larger for Models 3 and 4 (standard deviations 0.123 and 0.103) than for Models 1 and 2 (0.046). The worst case image matters here: Models 1 and 2 never fell below 0.746 and 0.761, whereas Models 3 and 4 had images at 0.123 and 0.000. Their lower mean scores are therefore influenced by a small number of failures, in addition to a generally lower level of agreement.

Figure 4.6 shows the per image distributions of Dice, IoU and HD95.

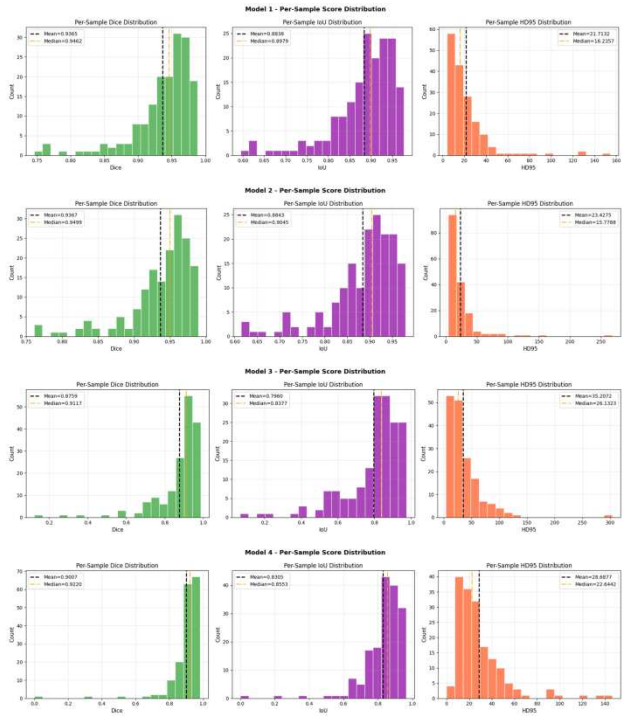


Figure 4.6: Per image Dice, IoU and HD95 distributions on the lesion containing test images (Models 1 to 4, top to bottom)

4.12 IoU Analysis

IoU follows the same order as Dice. For a single image IoU is a monotonic function of Dice, so the agreement is expected. The gap between Dice and IoU grows when per image scores vary more, which is visible for Model 3 (0.876 against 0.796).

4.13 Precision Analysis

Model 1 has the highest mean precision (0.9412), followed by Model 2 (0.9341). Model 4 has the lowest (0.8558). Precision is the fraction of predicted wound pixels that are correct, so a low value means over segmentation, here relative to the cropped reference mask. These are averages of per image ratios and not pooled pixel counts.

4.14 Recall Analysis

Model 4 has the highest recall (0.9703), and its interval [0.9550, 0.9803] lies above the other models' intervals. Combined with its low precision, this indicates that Model 4 tends to predict regions larger than the reference. Models 2 and 3 also have recall above precision. Model 1 is the most balanced (precision 0.9412, recall 0.9364). The paper attributes Model 4's high recall to its multi granularity design, but it can equally be a consequence of its loss, which adds a boundary weighted Dice term and auxiliary losses, or of training from scratch. The experiments cannot tell these apart.

4.15 HD95 Boundary Analysis

HD95 is measured in pixels of the cropped and resized network input (512 × 512 for Models 1, 2, 4 and 384 × 384 for Model 3). Models 1 and 2 have the smallest mean values (21.71 and 23.43). Model 3 is the largest (35.21). Two cautions apply. First, Model 3 works at a smaller image size, so a pixel there corresponds to a larger fraction of the wound crop than a pixel in the 512 pixel models, which

makes its HD95 not directly comparable with the others. Second, HD95 varies enormously between images: the standard deviation is between 0.8 and 1.0 times the mean for Models 1, 3 and 4, and about 1.2 times the mean for Model 2, whose worst image has HD95 of 268 pixels. Mean HD95 is therefore dominated by a few cases, and the ordering of Model 1 before Model 2 (difference 1.7 pixels) is well within the overlap of their intervals, [18.70, 25.21] and [19.49, 28.21].

The paper states that the HD95 ranking agrees with the Dice ranking. In fact the Model 1 / Model 2 order differs between the two (Model 2 first on Dice, Model 1 first on HD95), although the difference is small in both.

4.16 Confusion and Error Region Analysis

The notebooks do not record pixel level true positive, false positive and false negative counts for the full test set. The available evidence is the per image precision and recall of Table 4.3 and the qualitative difference maps of Section 4.28. From these, the pattern is: Models 1 and 2 are balanced with small errors in both directions, Model 4 errs toward over segmentation, and Models 3 and 4 have occasional large failures (low minimum scores).

4.17 Training Convergence

Table 4.5 summarises validation Dice at selected epochs, read from the training logs. Figures 4.3 and 4.4 show the training curves saved by the notebooks.

Table 4.5: Validation Dice at selected epochs (read from the training logs)

Epoch	Model 1	Model 2	Model 3	Model 4
1	0.854	0.621	0.395	0.535
5	0.880	0.920	0.788	0.800
10	0.647	0.933	0.812	0.858
20	0.925	0.929	0.843	0.895
30	0.930	0.938	0.868	0.896
50 (or last)	0.935	0.936 (epoch 34)	0.880	0.908

The pretrained models start higher than the models trained from scratch, Model 2 reaches 0.92 by epoch 5, and Model 3 is the slowest. Model 1's validation Dice at epoch 1 (0.854) is high, but the validation set includes empty mask images that score a perfect 1 when predicted empty (Section 4.5), so early values are partly inflated.

4.18 Validation Performance

The best validation Dice values (0.9357, 0.9406, 0.8811, 0.9111) order the models as Model 2, Model 1, Model 4, Model 3, the same order as on the test set. Validation values are not directly comparable with test values, for the reason in Section 4.5. Validation Dice of Models 1 and 2 is close to their test Dice, and for Models 3 and 4 it is also close (0.881 against 0.876 and 0.911 against 0.901), which gives no sign of a large gap between the two subsets, although the empty mask images in validation and the exclusion of such images in test work in opposite directions.

4.19 Test Time Augmentation Analysis

The reported test scores use six fold TTA, and the notebooks do not compute scores without TTA. How much the augmentation improved the scores is therefore **not measured** in this project. What can

be reported is the cost: TTA requires six forward passes, so the time per image is about six times the single pass latency (Section 4.23).

4.20 Empty Mask Test Image Analysis

The notebooks also compute scores on the full test set of 229 images, in which the 61 empty mask images are included (Table 4.6).

Table 4.6: Lesion only against full test set scores

Model	Dice (168 / 229)	IoU (168 / 229)	Precision (168 / 229)	Recall (168 / 229)	HD95 (168 / 229)
1	0.9365 / 0.9534	0.8838 / 0.9148	0.9412 / 0.9569	0.9364 / 0.9533	21.71 / 15.93
2	0.9367 / 0.9536	0.8843 / 0.9151	0.9341 / 0.9516	0.9451 / 0.9597	23.43 / 17.19
3	0.8759 / 0.9089	0.7960 / 0.8504	0.8748 / 0.9082	0.9001 / 0.9267	35.21 / 25.83
4	0.9007 / 0.9272	0.8305 / 0.8757	0.8558 / 0.8942	0.9703 / 0.9782	28.69 / 21.05

A simple calculation on these figures shows how the 61 empty mask images were scored. If the 168 lesion images have mean Dice L and the 61 empty images have mean Dice E, the full set mean is $F = (168L + 61E)/229$, so $E = (229F - 168L)/61$. For all four models this gives $E \approx 1.00$ (for example Model 1: $(229 \times 0.9534 - 168 \times 0.9365)/61 = 1.000$), and the same holds for IoU. For HD95, $(168 \times 21.7132)/229 = 15.93$, which is Model 1's full set value, so the empty images contributed an HD95 of exactly 0. *(derived)*

The full set improvement is therefore produced entirely by the empty mask images being scored as perfect. This means that **every one of the four models predicted an empty mask for every one of the 61 empty mask images**, with no false positive pixel at all. That is a surprisingly clean result, including for the weaker models, and it deserves an explanation. Two contributing facts are known from the code. First, the empty mask images are not cropped, while every labelled image is cropped around its reference mask. A network can therefore distinguish the two kinds of image from framing alone. Second, during training the empty mask images are presented with an empty target, and about 133 of them are unlabelled wound photographs. The networks may have learned the shortcut "uncropped, full frame image → output nothing". This is a hypothesis consistent with the numbers. It was not tested, for example by running the models on cropped versions of the 34 unlabelled test images. If it is correct, the models would be expected to miss the wounds in full frame photographs of real ulcers, which would be a serious restriction on practical use.

Because of the points in Sections 3.10.3 and 3.11, the full test set scores are given in this thesis only as a record of what the notebooks computed. They are not used for conclusions about segmentation quality. The paper's statement that the wound free images were "evaluated separately using classification based measures" is not what the notebooks did, and the thesis does not repeat it.

4.21 Confidence Interval Analysis

Ninety five per cent confidence intervals for the means were obtained by bootstrap resampling of the per image values of the 168 lesion containing test images (10,000 resamples, seed 42).

Table 4.7: Bootstrap 95% confidence intervals (168 lesion containing test images)

Model	Dice	IoU	Precision	Recall	HD95 (px)
1	0.9365 [0.9295, 0.9432]	0.8838 [0.8722, 0.8948]	0.9412 [0.9323, 0.9493]	0.9364 [0.9262, 0.9459]	21.71 [18.70, 25.21]
2	0.9367 [0.9296, 0.9433]	0.8843 [0.8725, 0.8954]	0.9341 [0.9234, 0.9440]	0.9451 [0.9355, 0.9539]	23.43 [19.49, 28.21]
3	0.8759 [0.8559, 0.8931]	0.7960 [0.7710, 0.8184]	0.8748 [0.8504, 0.8972]	0.9001 [0.8794, 0.9178]	35.21 [30.48, 40.60]
4	0.9007 [0.8833, 0.9146]	0.8305 [0.8108, 0.8478]	0.8558 [0.8387, 0.8717]	0.9703 [0.9550, 0.9803]	28.69 [25.46, 32.23]

The intervals describe the uncertainty from the finite number of test images. They do not include the uncertainty that arises from the training seed (only one run per model exists), from the choice of split, or from the possible presence of images of the same patient in training and test. The true uncertainty is therefore larger than the intervals suggest. Differences between Model 1 and Model 2 on every metric are small compared with their intervals. The differences between Models 1/2 and Model 4, and between Models 1/2 and Model 3, are larger than the interval widths for Dice, IoU and HD95.

4.22 Model Complexity Analysis

Table 4.8: Parameter counts by module (millions)

Module	Model 1	Model 2	Model 3	Model 4
Encoder	23.508 (ResNet 50)	87.564 (ConvNeXt B)	7.826 (U Net)	23.749 (stem + residual + Mamba stages)
Vision Mamba bottleneck	29.604	7.462	1.896	included in encoder
Node construction	0.550	0.268	0.279	0.196
Hypergraph reasoning	1.909	0.988	0.132 (fusion gate)	1.977
Symbolic / fusion module	0.131 + 0.919	none	included above	0.657
Decoder	5.108	3.290	3.164	9.173
Total	**61.729**	**99.573**	**13.296**	**35.753**

(Model 2 node construction is the sum of intensity 0.003, texture 0.034, position 0.034 and node fusion 0.197.)

Two observations follow. In Model 1 the Mamba style block with 2048 channels holds 29.6 M parameters, which is 48% of the model and more than the ResNet 50 encoder itself (23.5 M). The hypergraph and symbolic modules together hold under 3.5 M parameters in Model 1 and about 1 M in Model 2. The hypergraph modules are small parts of the models by parameter count, which does not tell us how much they matter for the outputs.

Table 4.9: FLOPs for one 512 × 512 input (thop, multiply accumulates doubled)

Model	GFLOPs
1	91.15
2	not measured
3	not measured (input would be 384 × 384)
4	111.56

The two missing values are left unmeasured because `thop` was not installed in those runs and the authors decided not to repeat them.

4.23 Computational Performance

Table 4.10: Inference latency in the experimental environment (single image, no TTA, Kaggle GPU)

Model	Latency (ms)	Throughput (images/s)	Six fold TTA, estimated (ms)
1	47.03	21.26	282.16
2	130.29	7.68	781.7 (derived: 6 × 130.29)
3	103.98	9.62	623.9 (derived: 6 × 103.98)
4	40.55	24.66	243.32

These are measurements in the Kaggle environment with a single image and one GPU. They are not a statement about real time clinical performance and were not obtained on a deployable device. TTA latency is an estimate, not a separate measurement. The timing protocol (number of repetitions, warm up, synchronisation) was not recorded.

Latency does not follow parameter count or FLOPs. Model 4 has more FLOPs than Model 1 (111.56 against 91.15 GFLOPs) yet a lower latency (40.55 against 47.03 ms), a difference of 14% that could be within the noise of an unreported timing protocol. Model 3 has the fewest parameters (13.3 M) but a latency of 104 ms, more than twice that of Model 1. A possible reason is that its operations on the node graph and on sampled positions are cheap in FLOPs but not in time, but this was not examined. For Model 2, 99.6 M parameters and 130 ms latency were the highest of the four.

Model 1 and Model 2 reached practically the same Dice, with Model 1 using 38% fewer parameters and about 2.8 times less time per image *(derived)*. This is a statement about these two designs in this environment and should not be generalised.

4.24 Hypergraph Analysis

Table 4.11 lists the structure of the hypergraphs. The first rows follow from the construction rules in the code (computed, not measured on images); the last row for Model 2 was measured.

Table 4.11: Hypergraph characteristics

Model	Nodes	Relation	Entries in adjacency matrix	Density	Mean degree
1	100	region (fixed)	3,450	0.345	34.5
1	100	boundary (fixed)	2,500	0.250	25.0
1	100	spatial, ±5 in node index (fixed)	1,070	0.107	10.7
1	100	colour and texture similarity	built per forward pass, not logged		
2	128	cosine similarity > 0.6	measured		94.55 of 127 possible
3	64	symbolic (fixed)	2,816	0.688	44.0
4	84	within level similarity plus cross level links	built per forward pass		cross level: 64 fine mid, 16 mid coarse

(Entries include the diagonal. The edge counts quoted in the paper for Model 1 , 1,675, 1,250 and 655 , are consistent with these entries counted as undirected pairs.)

The measurement for Model 2 deserves comment. Each node was connected, on average, to 94.55 of the 127 other nodes, which is 74%. A graph that is this dense behaves close to a near complete graph, in which a node's updated feature is close to an average over most of the other nodes. At a threshold of 0.6 the "dynamic" graph therefore selects relatively little. It is not clear from the notebook on which image or batch this degree was measured. It would also be useful to know how the degree changes with the threshold, which was not examined.

The Model 3 symbolic adjacency is also dense (density 0.69). Together with the facts of Section 3.13 (the hypergraph output reaches the decoder only as one averaged vector), this means the hypergraph modules act more like global context mixers than like selective relational structures.

Figure 4.7 shows the recorded Model 2 quantities: the magnitudes of the three node branches, their alignment with the fused representation, the hyperedge incidence matrix and the node degree histogram.

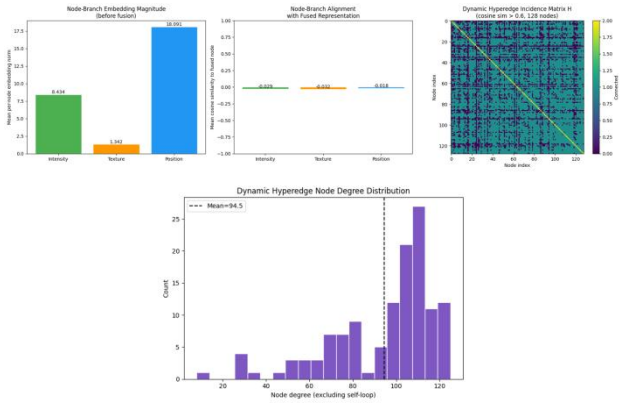


Figure 4.7: Model 2 node branch analysis, incidence matrix and node degree distribution (notebook output)

4.25 Symbolic Fusion Analysis

Table 4.12 collects the gate values recorded in the notebooks.

Table 4.12: Gate and fusion values

Model	Quantity	Value	Reference value at initialisation
1	Mean edge bed gate (Equation 3.2), validation batch	0.523	about 0.5 for a freshly initialised sigmoid gate
3	Mean edge bed gate in node embedding	0.762	about 0.5
3	Region fusion value α: bed / edge / perilesional / healthy	0.620 / 0.623 / 0.616 / 0.620	0.622 for all four
4	Cross scale gates: fine←mid / coarse←mid / mid←coarse / fine←mid again	0.465 / 0.463 / 0.365 / 0.354	about 0.5

A sigmoid gate whose inputs are small and whose weights are randomly initialised gives values near 0.5, so a mean value close to 0.5 does not by itself show that the gate has learned anything. Model 1's mean of 0.523 is of this kind. The paper's figure of the per node gate values shows that they are spread across a range of roughly 0.40 to 0.62 around this mean, which shows that individual gates do vary from node to node. That variation is evidence that the gate computes something that depends on the node; it is not evidence that the dependence matches any anatomical meaning, because the anatomical labels are index ranges (Section 3.18).

The values for Model 3 and Model 4 are discussed in the next two sections. No analysis in this project tests whether removing the symbolic rule changes the predictions, so the claim in the paper that symbolic constraints "contribute to structural consistency" is not supported by an experiment here.

Figure 4.8 shows the recorded gate values of Model 1.

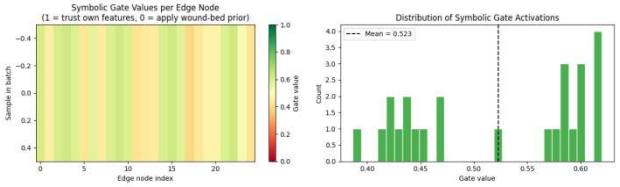


Figure 4.8: Model 1 symbolic gate values per edge node and their distribution (notebook output)

4.26 Model 3 Fusion Gate Analysis

The four region wise parameters of Model 3 started at the same value, $\sigma(0.5) = 0.622$ for every region. After 50 epochs (11,400 optimiser steps) they were 0.620 (wound bed), 0.623 (wound edge), 0.616 (perilesional) and 0.620 (healthy). In terms of the underlying parameter θ_r this is a movement of at most about 0.03 from the initial 0.5. *(derived)* The largest departure from the initial value is 0.006 in α.

What the measurement shows is that the four values did not separate from each other and also hardly moved from where they started. The paper's statement that "similar fusion weights were assigned across anatomical regions" is accurate, but the more precise statement is that the parameters stayed essentially at their initial values. The notebook prints the label "visual driven" for each region because every α is above 0.5. That label should not be read as a finding, since it is the initial condition, and it is not used in this thesis.

There are several possible reasons for the near constant values, and the present experiments do not discriminate between them: the gradient reaching these four scalars may be very small because the fused node features enter the decoder only after being averaged over all nodes; the scalars may simply receive few effective updates at a learning rate of 10^{-4}; or the visual and symbolic messages may be similar enough that their mixing proportion makes little difference to the loss. The paper adds lower input size and absence of pretraining as reasons. These factors may well have contributed to the lower score of Model 3, but their effect on the gate was not tested.

Note also that the notebook's text explaining the node embedding gate ("values consistently below 0.5 indicate …") contradicts its own printed value of 0.762, which is above 0.5. That printed explanation is not used here. A value of 0.762 means that edge nodes keep, on average, about three quarters of their own features and take about a quarter from the bed mean.

Figures 4.9 to 4.11 show the recorded Model 3 quantities.

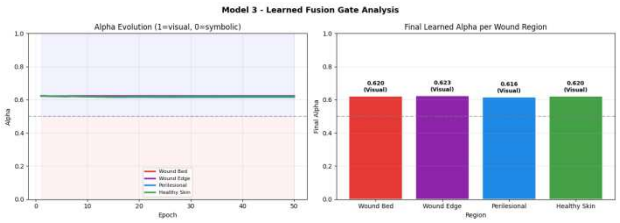


Figure 4.9: Model 3 fusion parameter α over the epochs and its final value per region

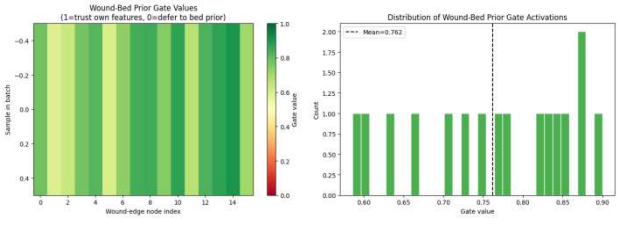


Figure 4.10: Model 3 wound bed prior gate values

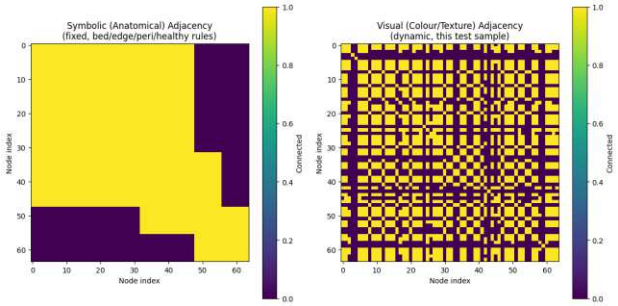


Figure 4.11: Model 3 symbolic and visual adjacency matrices for one sample

4.27 Model 4 Multi Granularity Analysis

The four cross scale gates of Model 4 average 0.465, 0.463, 0.365 and 0.354. The two gates that act in the top down direction (mid←coarse and the second fine←mid gate) are lower than the first two. A value below 0.5 means that, on average, a node moves less than halfway toward the mean of the neighbouring scale. All four values are within about 0.15 of 0.5, so the modification of node features is moderate, and nothing here shows that the cross scale exchange improved the segmentation. The notebook also produced node activation maps for the fine, mid and coarse levels (Figure 4.12); their norms were not printed, so no numerical analysis is given.

Model 4 is also the model with the highest recall and the lowest precision. It is natural to ask whether this is linked to the multi scale design. The loss includes a boundary weighted Dice term with weight 2 and three auxiliary losses, and the model was trained from scratch, so the link to the node scales is only one of several possibilities. None of them can be established from these experiments.

Figures 4.12, 4.13 and 4.14 show the recorded Model 4 quantities.

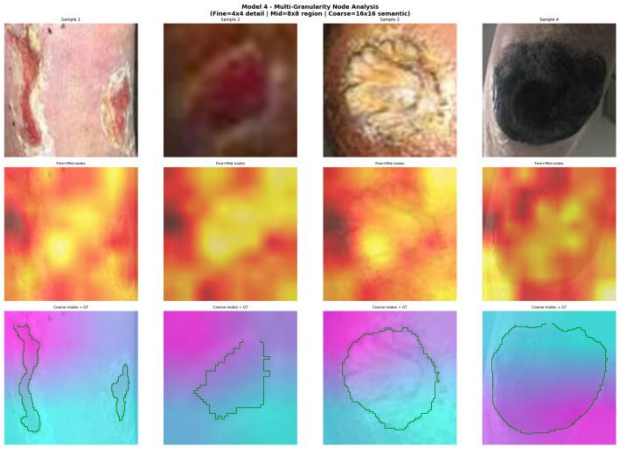


Figure 4.12: Model 4 node activation maps at the fine, mid and coarse levels for four samples (notebook output)

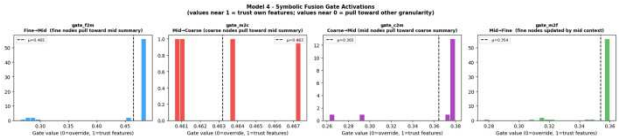


Figure 4.13: Model 4 cross scale gate activations (notebook output)

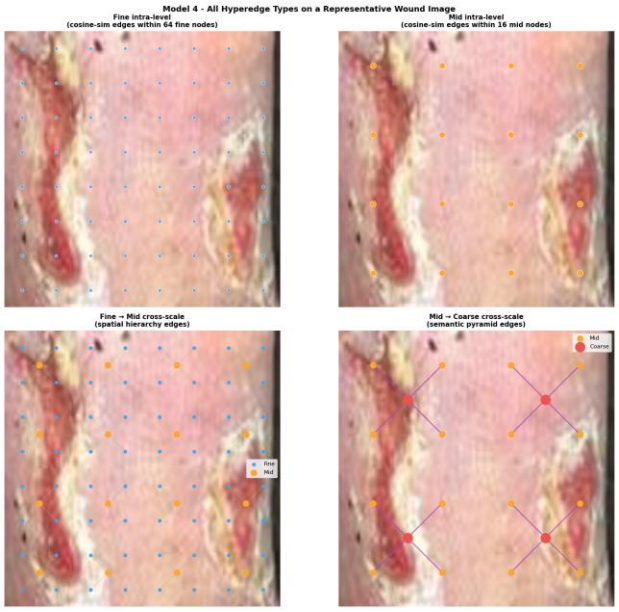


Figure 4.14: Model 4 hyperedge types on a representative wound image (notebook output)

4.28 Qualitative Segmentation Results

Figures 4.15 to 4.18 show predictions, reference masks and difference maps (true positive in green, false positive in red, false negative in blue) from the saved prediction grids. The same six test images are shown for every model. Models 1 and 2 produce smooth, compact masks that follow the main outline of the reference mask; the visible differences are thin red or blue rims at the border, and the reference masks have more irregular, often blocky edges than the predictions. Model 3 gives good outlines on several images but also shows the weaknesses seen in its metrics: on one image the prediction is a nearly rectangular region much larger than the wound. Model 4 gives smooth masks that are slightly larger than the reference on most images, which agrees with its high recall and lower precision, and on one image (a dark, necrotic looking wound) it predicted no wound at all, giving a Dice of 0.000. Because only a few examples are shown and the grids were not chosen at random, they illustrate typical behaviour only loosely and should not be taken as representative of the whole test set.

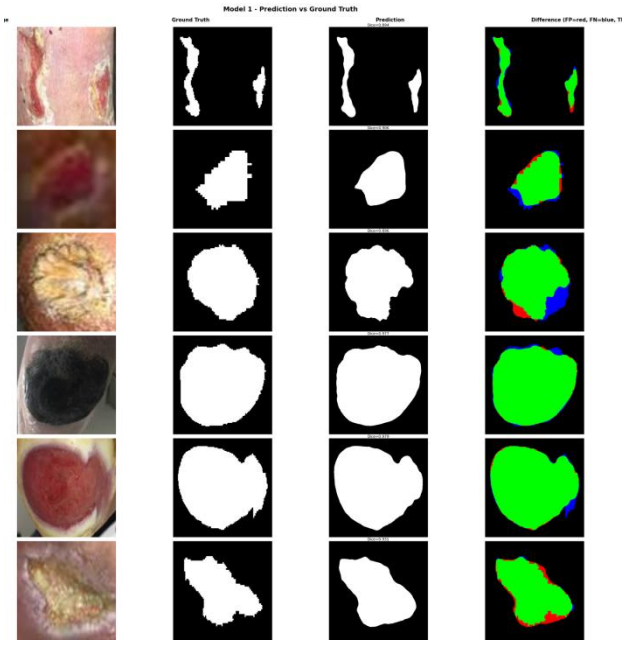


Figure 4.15: Model 1: input image, reference mask, prediction and difference map for six test images (TP green, FP red, FN blue)

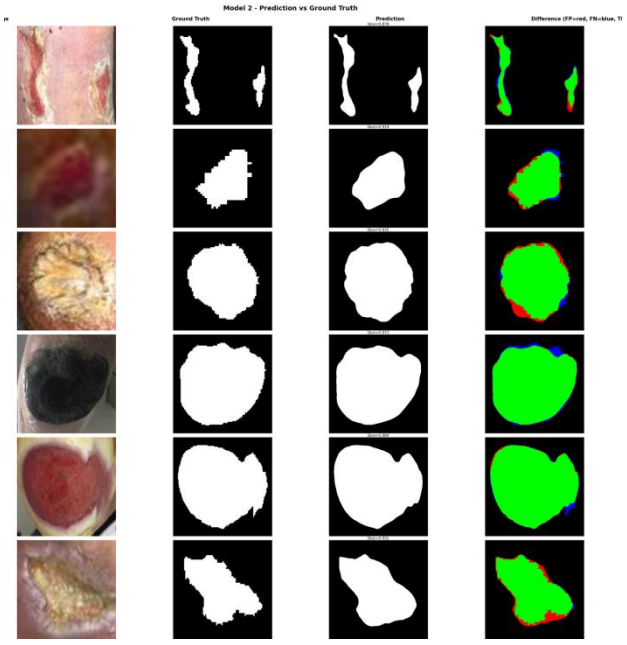


Figure 4.16: Model 2: input image, reference mask, prediction and difference map for six test images (TP green, FP red, FN blue)

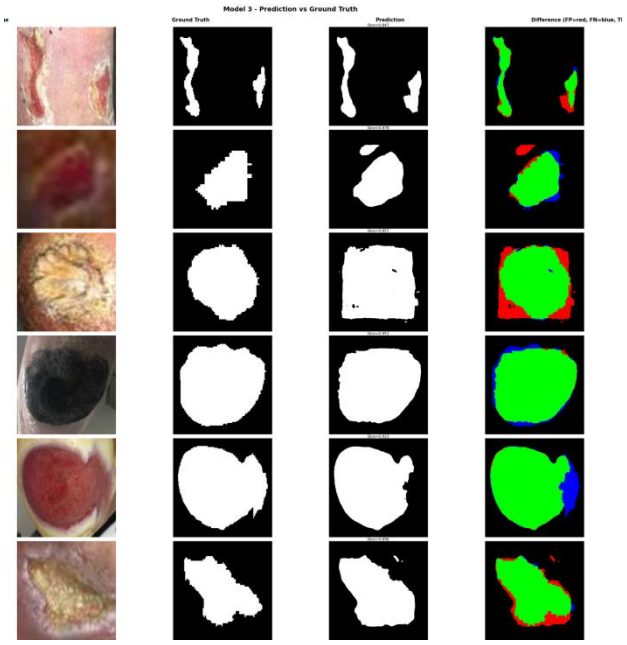


Figure 4.17: Model 3: input image, reference mask, prediction and difference map for six test images (TP green, FP red, FN blue)

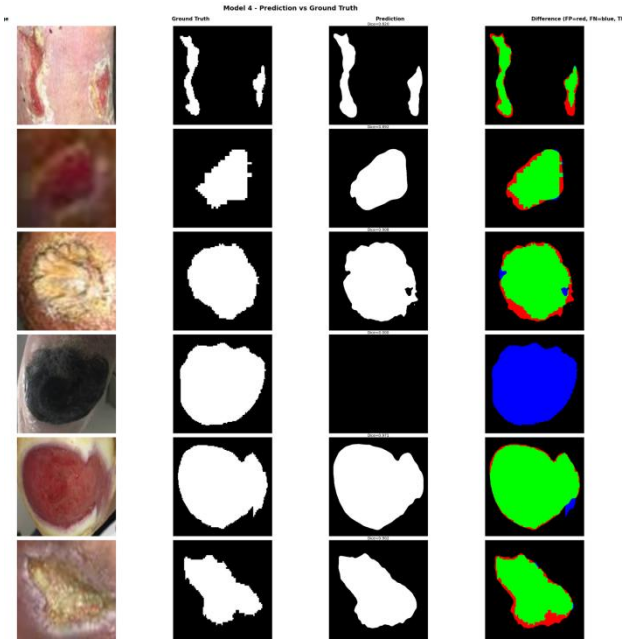


Figure 4.18: Model 4: input image, reference mask, prediction and difference map for six test images (TP green, FP red, FN blue)

4.29 False Positive Analysis

On images that contain a wound, over segmentation is the main weakness of Model 4, as shown by its precision of 0.856 against recall of 0.970. On the 61 empty mask images, no model produced a false positive at all (Section 4.20). Taken together, the models do not over segment where there is no reference wound but do tend to extend masks beyond the reference where there is one, at least for Model 4.

4.30 False Negative Analysis

The clearest false negative cases are the images with very low per image recall: Model 4's minimum recall is 0.0000 (an entirely missed wound), and Models 3 and 1 have minimum recalls of 0.108 and 0.615. Models 1 and 2 never missed more than about 40% of the reference pixels of any image. The images responsible for the lowest scores have not been inspected in this project.

Table 4.13: Error evidence available from the metrics and prediction grids

Model	Typical error type from metrics	Image level evidence available
1	balanced; small boundary differences	difference maps (Figure 4.15)
2	slight recall > precision; large outlier in HD95 (268 px)	difference maps (Figure 4.16)
3	wide spread; some very poor images (min Dice 0.123)	prediction grid only
4	over segmentation; at least one fully missed wound	prediction grid only

4.31 Architectural Comparison and Component Analysis

No controlled ablation was performed. The evidence on components is limited to the following, all of it descriptive.

Component	What was observed	What was not tested
Mamba style block	present in all models; large share of Model 1's parameters	any model without it
Hypergraph reasoning	graph structure and degree statistics (Table 4.11); gate values (Table 4.12)	any model without it
Edge bed symbolic rule	gate values; near initial values in Model 3	any model without it
Dynamic against fixed hyperedges	Models 1 and 2 equal in Dice	the same backbone with each
Multi granularity	Model 4 scores and gates	the same backbone with a single scale
Pretraining	the two pretrained models scored highest	any pair differing only in pretraining

A true ablation would need extra training runs, for example Model 1 without the Mamba style block, without the hypergraph path, and without the symbolic layer, repeated with several seeds. This is listed as future work.

4.32 Comparison with Existing Research

The paper compares its Models 1 and 2 with published results, and claims that they exceed all of them. This thesis does not repeat that claim, for the following reasons.

1. **Different test sets.** Published FUSeg results are normally scored on the official 200 image test set, whose masks are held by the organisers. That test set was not available here: its images are in the present pool without masks. The present test set is a random subset of 229 images from the pool.
2. **Different inputs.** The present networks receive wound centred crops made with the reference mask. Published segmentation methods operate on whole photographs.
3. **Different composition and protocol.** The pool includes Medetec images, patient level separation is not verifiable, and the metrics use smoothing and the HD95 conventions of Section 3.32.

4. **Mixed source values.** The comparison table of the project paper quotes some published scores for a different dataset from the one it names, and several of its citations do not match the cited text. Only values checked against the original publications are given in Table 4.14.

Table 4.14: Reported results in the literature and in this project (context only; no ranking is implied)

Study	Dataset and protocol	Dice (%)
Mahbod et al. [20]	Chronic wound dataset; ensemble of CNNs	92.07
Mahbod et al. [20]	FUSeg official test set	88.80
Dhar et al. [21]	Chronic wound dataset	92.70
Dhar et al. [21] (x FUSegNet)	FUSeg official test set	89.23
This project, Models 1 / 2	Pooled FUSC + Medetec, random test subset, wound centred crops, lesion containing images	93.65 / 93.67

The only conclusion supported is that scores of the order of 0.93 Dice are reached by the pretrained models under the protocol of this project. Whether they would be reached on whole photographs or on the official FUSeg test set is not known.

4.33 Discussion

The experiments produced a clear grouping: two pretrained models at about 0.937 Dice, which are indistinguishable from each other within the uncertainty of the test set, and two models trained from scratch, at 0.901 and 0.876, with much more variable per image scores and some outright failures. Whether the difference between these groups comes from pretraining, input size, loss, architecture, or a mixture, cannot be decided from the data. The most economical explanation, in the absence of controlled experiments, is the common one that ImageNet pretraining helps greatly with small medical datasets. It would be unwise to read the results as showing that Mamba style context, hypergraphs or symbolic rules were the decisive ingredients, since every model contains all of them and none was removed in a controlled test.

The internal measurements point to a modest role for the relational and symbolic parts in these implementations. The hypergraph output reaches the decoder as a single averaged vector. The Model 2 graph connects each node to about three quarters of the others. The Model 3 gates stayed at their initial values. The anatomical groups are node index ranges. These are all consistent with the networks relying mainly on their convolutional encoders and decoders for the segmentation. This is a hypothesis, to be tested by ablation, and not a conclusion.

The behaviour on empty mask images (a perfect score for every model) together with ground truth cropping and unlabelled training images suggests that the data pipeline may have shaped the models' behaviour more strongly than the architectures did. This is the most important caveat on all results in this chapter.

4.34 Limitations

- The evaluation is on a single pooled dataset and a single random split. No external or clinical test data were used, and the official FUSeg test set could not be used because its masks are private.
- The test inputs are cropped using the reference mask, so the task evaluated is easier than whole photograph segmentation.
- Patient level separation could not be independently verified from the available dataset structure.

- About 200 real wound photographs without reference masks were included as empty mask images, about 133 in training and 33 in validation.
- All models were trained once, with one seed. Differences smaller than seed to seed variation, such as the 0.0002 Dice difference between Models 1 and 2, carry no information.
- No baseline and no controlled ablation exist, so no statement about the contribution of any component is made.
- The Mamba style block is a simplified cumulative gating layer and not a state space recurrence.
- The anatomical groups are node index ranges and not derived from the image.
- The test model is the final EMA model, not the validation best checkpoint, and the validation scores include empty mask images.
- HD95 is in pixels of a cropped, resized input, is dominated by a few images, and takes the value 0 for images with an empty prediction and a non empty reference.
- FLOPs are missing for Models 2 and 3, TTA benefit was not measured, and the latency protocol was not recorded.
- Comparison with published results is limited by differences in data, protocol and unverified citations.

4.35 Threats to Validity

Internal validity. The shortcut possibility of Section 4.20, ground truth guided cropping, label noise in training and validation, and the lack of repeated runs weaken inferences about what the models learned.

Construct validity. Dice, IoU and HD95 on cropped, resized images are proxies for the clinically relevant quantity, which is wound area or boundary in physical units. No calibration of pixel size was available, so no measurement in millimetres can be derived.

External validity. The data come from two public collections whose acquisition, patient population and annotation are only partly documented. The results may not transfer to other cameras, other wound types or other settings.

Statistical validity. Bootstrap intervals cover test sample variation only. No paired tests were done.

CHAPTER 5
CONCLUSION AND FUTURE WORK

5.1 Summary of the Work

The project set out to build and compare deep learning models that outline diabetic foot ulcers in clinical photographs. Four architectures were implemented in PyTorch. Each combines an image encoder, a Mamba style context block, a hypergraph module over feature nodes and a convolutional decoder. They differ in the encoder (ResNet 50, ConvNeXt B, a small U Net encoder and a custom ConvNeXt style encoder), in how nodes and hyperedges are built, in the loss function, and in whether a fixed structural rule about wound regions is applied (Chapter 3).

The models were trained on a pooled set of 1,370 images from the Foot Ulcer Segmentation Challenge (FUSC) and Medetec collections. The split produced by the code was 913 training, 228 validation and 229 test images, which is 66.6 : 16.6 : 16.7 per cent and not the 70 : 15 : 15 stated in the project paper. All four models used the same optimiser, schedule, exponential moving average, gradient clipping and six fold test time augmentation. They were evaluated with Dice, IoU, precision, recall and HD95 on the 168 test images whose reference mask contains a wound (Chapter 4).

5.2 Major Findings

Segmentation accuracy. On the 168 lesion containing test images, Model 2 obtained the highest Dice (0.9367) and IoU (0.8843). Model 1 obtained the highest precision (0.9412) and the lowest HD95 (21.71 pixels). Model 4 obtained the highest recall (0.9703). Model 3 had the lowest Dice (0.8759) and the highest HD95 (35.21 pixels). No model led on every metric.

Models 1 and 2 are tied. Their Dice differs by 0.0002, while the 95% bootstrap interval on each is about 0.014 wide. The experiments cannot separate them.

Two groups. The two models with ImageNet pretrained encoders (Models 1 and 2) scored about 0.937 Dice, and the two trained from scratch scored 0.901 (Model 4) and 0.876 (Model 3), with larger per image spread and some outright failures, such as a Dice of 0.0 on at least one Model 4 image. This is consistent with pretraining being an important factor, but the models also differ in input size, loss, node construction and decoder, so pretraining could not be separated from these.

Computational cost. Model 1 has 61.7 M parameters and 91.15 GFLOPs, and Model 4 has 35.8 M parameters and 111.56 GFLOPs. FLOPs were not measured for Models 2 (99.6 M parameters) and 3 (13.3 M parameters). Single image latency without test time augmentation in the Kaggle environment was 47.03 ms (Model 1), 130.29 ms (Model 2), 103.98 ms (Model 3) and 40.55 ms (Model 4). Model 1 matched Model 2 in Dice with fewer parameters and lower latency. Latency did not follow parameter count: Model 3, the smallest, was slower than Model 1.

Internal quantities. The four region wise fusion parameters of Model 3 stayed essentially at their initial value of 0.622 (final values 0.616 to 0.623), so they did not separate by region. Model 2's cosine similarity hypergraph connected each node to about 74% of the others on average. In every model the hypergraph output reaches the decoder as one averaged vector per image (three in Model 4). These measurements suggest that the hypergraph and symbolic parts act mainly as global context mixers. This is a hypothesis for later testing and not a finding about their contribution.

Properties of the data pipeline. Analysis of the code showed five points that limit how far the scores can be interpreted: the 200 FUSC test folder images, whose masks the challenge organisers withheld, were loaded as all zero masks and are spread across the splits (about 133 in training, 33 in validation, 34 in test); every labelled image was cropped around its own reference mask, including at test time;

the split was stratified only by whether a folder has labels, and patient level separation could not be verified; the final test model was the end of training EMA model and not the validation best checkpoint; and the metrics score an empty prediction on an empty reference as perfect. All four models predicted an empty mask for all 61 empty mask test images, which, together with the cropping, suggests a possible framing shortcut that was not tested.

5.3 Achievement of Objectives

Objective	Status
1. Develop an automated DFU segmentation framework	Achieved: four trained and evaluated models
2. Incorporate a Mamba style context block	Achieved, with the qualification that the block is a simplified gated cumulative average and not a state space recurrence (Section 3.15)
3. Investigate hypergraph reasoning on node features	Achieved as implementation and description. Its contribution to accuracy was not isolated
4. Incorporate a structural prior into selected models	Implemented in Models 1 and 3 as node index groups with a gating rule, and a cross scale gate in Model 4. The groups are not aligned to image content, and the benefit was not tested
5. Develop and compare four variants under one protocol	Achieved. The comparison is of whole designs and not controlled
6. Evaluate with overlap and boundary metrics	Achieved, with the caveats on cropping, empty masks and HD95 conventions
7. Analyse computational characteristics	Achieved in part: FLOPs missing for Models 2 and 3, TTA benefit and training time not measured
8. Analyse gate values and graph structure	Achieved where the notebooks recorded them
9. Identify strengths and limitations	Achieved (Sections 4.35 and 4.36)

5.4 Answers to Research Questions

RQ1. How accurately do the four architectures segment DFU regions in the pooled dataset? Within the protocol used, the two pretrained models reached about 0.937 Dice and 0.884 IoU on the lesion containing test images, Model 4 reached 0.901 Dice and Model 3 reached 0.876. These scores describe segmentation inside wound centred crops made with the reference mask, so they are not estimates of accuracy on uncropped photographs.

RQ2. How do the architectures differ across the five metrics? Model 2 leads on Dice and IoU, Model 1 on precision and HD95, and Model 4 on recall. Models 1 and 2 are within noise of each other on every metric. Model 4 over segments relative to the reference (precision 0.856, recall 0.970), and Models 3 and 4 have the widest per image spread.

RQ3. What do the recorded hypergraph and symbolic quantities show? They show that the Model 3 gate parameters did not move from their initial values, that the Model 2 graph is dense, that the hypergraph reaches the decoder only as an averaged vector, and that Model 1's edge bed gate varies from node to node without any demonstrated link to wound anatomy. None of the analyses tests whether removing a component changes the output, so they do not show that the components helped.

RQ4. What computational trade offs exist? Parameter count, FLOPs and latency did not rank the models the same way. The two best scoring models differ greatly in cost: Model 1 used fewer parameters and less time than Model 2 for the same Dice. Model 4 was the fastest in single pass latency. Six fold TTA multiplies the per image time by about six.

RQ5. What does the experimental procedure allow the results to show? It allows a comparison of four designs on a random held out subset of a pooled public dataset, with a single run per model. It does not allow claims about whole photograph segmentation, ranking against published leaderboard results, the effect of any individual component, or clinical use.

5.5 Contributions

1. Implementation of four segmentation architectures that combine a convolutional encoder, a Mamba style context block and hypergraph reasoning with different node constructions, hyperedge definitions, decoders and losses.
2. A comparison of the four under one training and evaluation protocol, with bootstrap confidence intervals, parameter counts, FLOPs where measured, and latency.
3. A record of internal quantities (gate values, graph statistics, parameter shares) and an assessment of what they do and do not show.
4. A critical account of the data pipeline and evaluation procedure, with its consequences for interpreting the results.

5.6 Practical Significance

The results show that networks of this kind, with a pretrained encoder, can reproduce reference outlines closely when given a wound centred crop of a photograph. That is a limited statement. A system that could support wound measurement would need to find the wound in an uncropped photograph, which this project did not evaluate, and to convert pixel measurements to physical units, which was not possible with the available data. The work is best read as an exploratory architectural study and as a worked example of checks that a segmentation pipeline should undergo before its scores are trusted. It is not evidence of clinical effectiveness.

5.7 Limitations

- The evaluation uses a single pooled dataset and one random split. No external or clinical data were used, and the official FUSeg test set was unavailable because its masks are private.
- Test images were cropped around the reference mask, so the task evaluated is easier than whole photograph segmentation.
- About 200 real wound photographs without masks were included as empty mask images, which adds label noise to training and validation.
- Patient level separation across splits could not be verified.
- Each model was trained once, so differences smaller than seed to seed variation, such as that between Models 1 and 2, carry no information.
- No baseline and no controlled ablation were run, so no component's contribution is established. The models differ in several respects at once.
- The Mamba style block is a simplified cumulative gating layer, and the anatomical groups are node index ranges not tied to the image.
- HD95 is in pixels of a cropped and resized input, is dominated by a few images, and is recorded as 0 when a prediction is empty and the reference is not.
- FLOPs are missing for two models, the effect of test time augmentation was not measured, and the latency protocol was not recorded.
- Comparison with published results is limited by differences in data and protocol.

5.8 Future Work

The following follow directly from the limitations above.

1. **Clean the data.** Remove the images without reference masks from the supervised pipeline, use a split that is stratified by wound size and source, and, if patient identifiers can be obtained, split by patient.
2. **Remove the mask guided crop.** Evaluate on uncropped photographs, and add a detection or localisation stage if needed, as in detect then segment designs [22].
3. **Controlled ablation.** Train versions of one model without the context block, without the hypergraph path, and without the symbolic rule, and a pair that differs only in pretraining, each with several seeds and a paired statistical test.
4. **Make the prior anatomical.** Derive node groups from the image or from a coarse segmentation, so that the structural rule refers to real regions.
5. **Replace the context block with a true selective state space layer** with multi directional scanning, and compare it with the simplified block.
6. **Cost reduction.** Measure the benefit of test time augmentation, reduce its overhead, and examine knowledge distillation and compression, given the latencies recorded here.
7. **External validation.** Evaluate on other wound datasets and, with ethical approval, on clinical data from the intended setting, with pixel to millimetre calibration.
8. **Ensembles and uncertainty.** Combining several models and estimating the uncertainty of each prediction are left as open directions.
9. **Longitudinal use.** Study whether outlines from repeated photographs of the same wound are consistent over time.

5.9 Final Conclusion

Four neurosymbolic style segmentation networks were built, trained and evaluated for diabetic foot ulcer outlining. With pretrained encoders, two of them reached about 0.937 Dice on wound centred crops, and the other two, trained from scratch, scored lower and varied more. The recorded gate and graph measurements point to a limited, global role for the hypergraph and symbolic parts, but no experiment isolates their effect. Examination of the data pipeline showed that the results come from a protocol that is easier than whole photograph segmentation and that contains label noise. The scores should be read within that protocol. Closing these gaps is the main route by which this work could be turned into a result that others can compare against and build on.

REFERENCES

[1] Ronneberger, O., Fischer, P., Brox, T.: U Net: convolutional networks for biomedical image segmentation. MICCAI 2015, LNCS 9351, pp. 234 to 241

[2] Zhou, Z., Rahman Siddiquee, M.M., Tajbakhsh, N., Liang, J.: UNet++: a nested U Net architecture for medical image segmentation. DLMIA 2018, LNCS 11045, pp. 3 to 11

[3] Oktay, O. et al.: Attention U Net: learning where to look for the pancreas. MIDL 2018 (arXiv:1804.03999)

[4] Schlemper, J. et al.: Attention gated networks: learning to leverage salient regions in medical images. Med. Image Anal. 53, 197 to 207 (2019)

[5] Chen, J., Lu, Y., Yu, Q., Luo, X., Adeli, E., Wang, Y., Lu, L., Yuille, A.L., Zhou, Y.: TransUNet: transformers make strong encoders for medical image segmentation. arXiv:2102.04306 (2021)

[6] Cao, H. et al.: Swin Unet: Unet like pure transformer for medical image segmentation. ECCV 2022 Workshops

[7] Dosovitskiy, A. et al.: An image is worth 16x16 words: transformers for image recognition at scale. ICLR 2021

[8] Gu, A., Goel, K., Ré, C.: Efficiently modeling long sequences with structured state spaces. ICLR 2022

[9] Gu, A., Dao, T.: Mamba: linear time sequence modeling with selective state spaces. COLM 2024

[10] Zhu, L. et al.: Vision Mamba: efficient visual representation learning with bidirectional state space model. ICML 2024

[11] Liu, Y. et al.: VMamba: visual state space model

[12] Ma, J., Li, F., Wang, B.: U Mamba: enhancing long range dependency for biomedical image segmentation. arXiv 2024

[13] Ruan, J., Li, J., Xiang, S.: VM UNet: Vision Mamba UNet for medical image segmentation. arXiv:2402.02491 (2024)

[14] Kipf, T.N., Welling, M.: Semi supervised classification with graph convolutional networks. ICLR 2017

[15] Feng, Y., You, H., Zhang, Z., Ji, R., Gao, Y.: Hypergraph neural networks. AAAI 33(1), 3558 to 3565 (2019)

[16] Antelmi, A. et al.: A survey on hypergraph representation learning. ACM Comput. Surv. 56(1), 1 to 38 (2023)

[17] d'Avila Garcez, A., Lamb, L.C.: Neurosymbolic AI: the 3rd wave. Artif. Intell. Rev. 56 (2023)

[18] Wang, C., Mahbod, A., Ellinger, I., Galdran, A., Gopalakrishnan, S., Niezgoda, J., Yu, Z.: FUSeg: the Foot Ulcer Segmentation Challenge. Information 15(3), 140 (2024). doi:10.3390/info15030140

[19] Wang, C. et al.: Fully automatic wound segmentation with deep convolutional neural networks. Sci. Rep. 10 (2020)

[20] Mahbod, A., Schaefer, G., Ecker, R., Ellinger, I.: Automatic foot ulcer segmentation using an ensemble of convolutional neural networks. ICPR 2022, pp. 4358–4364. doi:10.1109/ICPR56361.2022.9956253

[21] Dhar, M.K., Zhang, T., Patel, Y., Gopalakrishnan, S., Yu, Z.: FUSegNet: a deep convolutional neural network for foot ulcer segmentation. Biomed. Signal Process. Control (2024); arXiv:2305.02961

[22] Scebba, G. et al.: Detect and Segment: a deep learning approach to automate wound image segmentation. Inform. Med. Unlocked (2022); arXiv:2111.01590

[23] Ramachandram, D. et al.: Fully automated wound tissue segmentation using deep learning on mobile devices: cohort study. JMIR mHealth uHealth 10(4), e36977 (2022)

[24] Asare, A. et al.: FUTransUNet GradCAM: a hybrid Transformer U Net with self attention and explainable visualizations for foot ulcer segmentation. arXiv:2508.03758 (2025)

[25] Alabdulhafith, M. et al.: Automated wound care by employing a reliable U Net architecture combined with ResNet feature encoders for monitoring chronic wounds. Front. Med. (2024)

[26] Hamrani, A. et al.: Beyond labels: zero shot diabetic foot ulcer wound segmentation with self attention diffusion models. arXiv:2504.17628 (2025)

[27] Zhou, G.-X. et al.: Construction and validation of a deep learning based diagnostic model for segmentation and classification of diabetic foot. Front. Endocrinol. (2025)

[28] Yap, M.H. et al.: Deep learning in diabetic foot ulcers detection: a comprehensive evaluation. Comput. Biol. Med. 135, 104596 (2021)

[29] Milletari, F., Navab, N., Ahmadi, S.-A.: V Net: fully convolutional neural networks for volumetric medical image segmentation. 3DV 2016

[30] Lin, T.-Y. et al.: Focal loss for dense object detection. ICCV 2017

[31] Berman, M., Rannen Triki, A., Blaschko, M.B.: The Lovász softmax loss. CVPR 2018

[32] Huttenlocher, D.P., Klanderman, G.A., Rucklidge, W.J.: Comparing images using the Hausdorff distance. IEEE TPAMI 15(9), 850 to 863 (1993)

[33] Taha, A.A., Hanbury, A.: Metrics for evaluating 3D medical image segmentation: analysis, selection, and tool. BMC Med. Imaging 15, 29 (2015)

[34] Aydin, O.U. et al.: On the usage of average Hausdorff distance for segmentation performance assessment: hidden error when used for ranking. Eur. Radiol. Exp. 5, 4 (2021)

[35] He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. CVPR 2016

[36] Liu, Z. et al.: A ConvNet for the 2020s. CVPR 2022

[37] Lin, T.-Y. et al.: Feature pyramid networks for object detection. CVPR 2017

[38] Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. ICLR 2019

[39] Tomar, D. et al.: OptTTA: learnable test time augmentation for source free medical segmentation. MIDL 2022

Appendix A: Dataset Description

The experiments used two public sources of wound photographs, both obtained from one Kaggle collection named “wound seg 3 datasets”. The Foot Ulcer Segmentation Challenge (FUSeg) set has 810 training, 200 validation and 200 test photographs from 889 patients [18]; the annotations of its 200 test images are withheld by the challenge organisers, so in this project those images were loaded without reference masks. The Medetec subset used here has 152 training and 8 test images with masks. The folder structure and counts are given in Table A.1. Section 3.3 to 3.7 describe how the data were read and combined.

Table A.1: Dataset composition as read by the notebooks

Source	Folder	Images	Masks available
FUSC	train	810	yes
FUSC	validation	200	yes
FUSC	test	200	**no (withheld by organisers)**
Medetec	train	152	yes
Medetec	test	8	yes
Total		**1,370**	1,170 with masks

Table A.2: Dataset distribution (images and reference masks)

Source	Folder	Images	Reference masks
FUSC	train	810	810
FUSC	validation	200	200
FUSC	test	200	none (withheld by the challenge organisers)
Medetec	train	152	152
Medetec	test	8	8
Total		1,370	1,170

Appendix B: Dataset Statistics

The pooled 1,370 images were divided with a two stage stratified split (seed 42). The last column of Table B.1 was obtained by re running the split logic and not from a notebook output (Section 3.9). The test subset contains 168 images whose reference mask has a wound and 61 images whose mask is empty, of which 34 come from the folder without labels (Section 3.11).

Table B.1: Dataset split produced by the implementation

Subset	Images	Share of 1,370	Images from the unlabelled FUSC folder
Training	913	66.6%	133
Validation	228	16.6%	33
Test	229	16.7%	34

Appendix C: Model Architecture Details

Table C.1 compares the four architectures component by component. Table C.2 lists where the Mamba inspired block acts, Table C.3 the input and node configuration, and Table C.4 the node index ranges used as anatomical groups in Models 1 and 3.

Table C.1: Model architecture comparison

Component	Model 1	Model 2	Model 3	Model 4
Encoder	ResNet 50 (ImageNet)	ConvNeXt B (ImageNet)	3 stage U Net (scratch)	ConvNeXt style stem + residual/Mamba stages (scratch)
Input size	512	512	384	512
Nodes	100	128	64	64 + 16 + 4 = 84
Hyperedges	five relation types (3 fixed, 2 similarity)	one similarity based, rebuilt each pass	visual + symbolic	within level similarity + cross level links
Reasoning layers	3 attention layers per relation, shared	3 spectral HGNN layers	one fusion step	3 bottom up + 3 top down layers
Anatomical rule	edge bed gating	none	edge bed gating + symbolic adjacency	cross scale gating only
Decoder	U Net	FPN	U Net	U Net style with deep supervision

Table C.2: Placement of the Vision Mamba block

Model	Where the Vision Mamba block acts	Channels
1	Final ResNet 50 stage (stride 32)	2048
2	Final ConvNeXt B stage (stride 32)	1024
3	U Net bottleneck	512
4	After each of three encoder stages and after the last stage	192, 384, 768, 768

Table C.3: Model specific input and node configuration

Model	Input	Nodes	Node dimension
1	512 × 512	100	256
2	512 × 512	128	256
3	384 × 384	64	256
4	512 × 512	84 (64 + 16 + 4)	256

Table C.4: Node index ranges used as anatomical groups

Model	Wound bed	Wound edge	Perilesional	Healthy
1 (100 nodes)	0 to 49	50 to 74	75 to 89	90 to 99
3 (64 nodes)	0 to 31	32 to 47	48 to 55	56 to 63

Appendix D: Hyperparameter Configuration

Table D.1: Common training configuration

Setting	Value
Optimiser	AdamW, weight decay 10^{-4}
Initial learning rate	1×10^{-4}
Warm up	5 epochs, linear
Schedule after warm up	cosine annealing to 0 over the 50 epoch horizon
Batch size	4
Maximum epochs	50
EMA decay	0.999
Gradient clipping	max norm 1.0
Early stopping	patience 10 epochs on validation Dice IoU mean
Seed	42

Table D.2: Model specific loss functions

Model	Loss
1	0.4 BCE + 0.6 Dice
2	0.4 Focal ($\gamma = 2$, $\alpha = 0.25$) + 0.6 Dice
3	0.5 Lovász hinge + 0.5 BCE
4	0.5 BCE + 0.5 (Dice + 2 × boundary Dice), plus 0.4/0.3/0.2 auxiliary terms

Table D.3: Preprocessing operations (all four models)

Step	Operation	Applied to
1	Read RGB image; read mask and binarise at 127	all images
2	Crop to the mask's bounding box plus 10 pixels	images with a mask file
3	Resize to a fixed square (bilinear for images, nearest for masks)	all images
4	Normalise with ImageNet mean and standard deviation	image only

Table D.4: Training augmentation operations

Operation	Probability / range
Horizontal flip	0.5
Vertical flip	0.5
Rotation by 0, 90, 180 or 270 degrees	one of four chosen uniformly
Elastic deformation (smoothed random displacement field, grid scale 20)	0.3
Brightness, contrast, saturation jitter (each factor 0.8 to 1.2)	0.5

Appendix E: Mathematical Formulations

Table E.1 collects the equations that define the implemented components, losses and weight averaging. The evaluation metrics are defined in Table E.2 (TP, FP and FN are the true positive, false positive and false negative pixel counts of one image; a constant 1 is added to numerator and denominator).

Table E.1: Equations used in the thesis

Quantity	Equation	Section
Mamba inspired gated cumulative context block	$y_t = D \odot x_t + \frac{1}{L}\sum_{s \le t} x_s \odot \sigma(\Delta_s)$	3.15
Edge bed symbolic gate	$X'_{\text{edge}} = g \odot X_{\text{edge}} + (1-g) \odot \bar{X}_{\text{bed}}, \quad g = \sigma\left(W\left[X_{\text{edge}}; \bar{X}_{\text{bed}}\right] + b\right)$	3.19
Dynamic hypergraph adjacency (Model 2)	$H_{ij} = \mathbb{1}\left[\frac{\bar{z}_i \cdot \bar{z}_j}{\| \bar{z}_i \| \, \| \bar{z}_j \|} > \tau\right], \quad \tau = 0.6$	3.21
Region wise visual symbolic fusion (Model 3)	$X' = \mathrm{LN}\left(X + \alpha_r \, m_{\text{vis}} + (1-\alpha_r)\, m_{\text{sym}}\right), \quad \alpha_r = \sigma(\theta_r)$	3.22
Dice loss	$\mathcal{L}_{\text{Dice}} = 1 - \frac{2\sum_i p_i \, g_i + \epsilon}{\sum_i p_i + \sum_i g_i + \epsilon}$	3.24
Binary cross entropy loss	$\mathcal{L}_{\text{BCE}} = -\frac{1}{n}\sum_i [g_i \log p_i + (1-g_i)\log(1-p_i)]$	3.24
Focal loss	$\mathcal{L}_{\text{focal}} = \frac{1}{n}\sum_i \alpha \, (1-p_{t,i})^{\gamma} \, \mathrm{BCE}_i$	3.24
Boundary weighted Dice loss (Model 4)	$\mathcal{L}_{\text{bDice}} = \mathcal{L}_{\text{Dice}} + 2\,\mathcal{L}_{\text{Dice}}^{\text{band}}$	3.24
Deep supervision total loss (Model 4)	$\mathcal{L} = \ell(\hat{y}_0) + 0.4\,\ell(\hat{y}_1) + 0.3\,\ell(\hat{y}_2) + 0.2\,\ell(\hat{y}_3), \quad \ell = 0.5\,\mathrm{BCE} + 0.5\,\mathcal{L}_{\text{bDice}}$	3.24
Exponential moving average of weights	$\theta^{\text{EMA}} \leftarrow 0.999\,\theta^{\text{EMA}} + 0.001\,\theta$	3.27

Table E.2: Evaluation metrics as implemented

Metric	Definition as implemented
Dice	$(2\,TP + 1)/(2\,TP + FP + FN + 1)$
IoU	$(TP + 1)/(TP + FP + FN + 1)$
Precision	$(TP + 1)/(TP + FP + 1)$
Recall	$(TP + 1)/(TP + FN + 1)$
HD95	95th percentile of pooled boundary to boundary distances, in pixels

Appendix F: Training Records

The training curves recorded by the notebooks are shown in Chapter 4. Table F.1 gives the training and validation summary of the four runs and Table F.2 the validation Dice at selected epochs.

Table F.1: Training and validation summary

Model	Epochs run	Epoch of best score	Best val Dice	Best val IoU	Best val combined	Early stop
1	50 / 50	48	0.9357	0.8893	0.9125	no
2	34 / 50	24	0.9406	0.8951	0.9178	yes
3	50 / 50	45	0.8811	0.8131	0.8471	no
4	50 / 50	41	0.9111	0.8508	0.8809	no

Table F.2: Validation Dice at selected epochs (read from the training logs)

Epoch	Model 1	Model 2	Model 3	Model 4
1	0.854	0.621	0.395	0.535
5	0.880	0.920	0.788	0.800
10	0.647	0.933	0.812	0.858
20	0.925	0.929	0.843	0.895
30	0.930	0.938	0.868	0.896
50 (or last)	0.935	0.936 (epoch 34)	0.880	0.908

Appendix G: Test Set Results

Table G.1: Test set performance on the 168 lesion containing images (EMA weights, six fold TTA)

Model	Dice	IoU	Precision	Recall	HD95 (px)
1	0.9365 (0.0456; 0.7459 to 0.9885)	0.8838 (0.0751; 0.5948 to 0.9773)	0.9412 (0.0557; 0.7024 to 1.0000)	0.9364 (0.0657; 0.6147 to 0.9974)	21.71 (21.53; 4.00 to 154.64)
2	0.9367 (0.0461; 0.7612 to 0.9894)	0.8843 (0.0763; 0.6145 to 0.9789)	0.9341 (0.0672; 0.6710 to 1.0000)	0.9451 (0.0609; 0.6166 to 0.9987)	23.43 (28.79; 4.12 to 268.43)
3	0.8759 (0.1233; 0.1229 to 0.9875)	0.7960 (0.1569; 0.0655 to 0.9753)	0.8748 (0.1536; 0.1422 to 0.9993)	0.9001 (0.1263; 0.1082 to 1.0000)	35.21 (33.01; 3.61 to 303.71)
4	0.9007 (0.1027; 0.0000 to 0.9819)	0.8305 (0.1225; 0.0000 to 0.9645)	0.8558 (0.1087; 0.1974 to 1.0000)	0.9703 (0.0851; 0.0000 to 1.0000)	28.69 (22.44; 0.00 to 146.37)

Table G.2: Bootstrap 95% confidence intervals (168 lesion containing test images)

Model	Dice	IoU	Precision	Recall	HD95 (px)
1	0.9365 [0.9295, 0.9432]	0.8838 [0.8722, 0.8948]	0.9412 [0.9323, 0.9493]	0.9364 [0.9262, 0.9459]	21.71 [18.70, 25.21]
2	0.9367 [0.9296, 0.9433]	0.8843 [0.8725, 0.8954]	0.9341 [0.9234, 0.9440]	0.9451 [0.9355, 0.9539]	23.43 [19.49, 28.21]
3	0.8759 [0.8559, 0.8931]	0.7960 [0.7710, 0.8184]	0.8748 [0.8504, 0.8972]	0.9001 [0.8794, 0.9178]	35.21 [30.48, 40.60]
4	0.9007 [0.8833, 0.9146]	0.8305 [0.8108, 0.8478]	0.8558 [0.8387, 0.8717]	0.9703 [0.9550, 0.9803]	28.69 [25.46, 32.23]

Table G.3: Lesion only against full test set scores

Model	Dice (168 / 229)	IoU (168 / 229)	Precision (168 / 229)	Recall (168 / 229)	HD95 (168 / 229)
1	0.9365 / 0.9534	0.8838 / 0.9148	0.9412 / 0.9569	0.9364 / 0.9533	21.71 / 15.93
2	0.9367 / 0.9536	0.8843 / 0.9151	0.9341 / 0.9516	0.9451 / 0.9597	23.43 / 17.19
3	0.8759 / 0.9089	0.7960 / 0.8504	0.8748 / 0.9082	0.9001 / 0.9267	35.21 / 25.83
4	0.9007 / 0.9272	0.8305 / 0.8757	0.8558 / 0.8942	0.9703 / 0.9782	28.69 / 21.05

Appendix H: Error Evidence

The prediction grids for the first six test images are shown in Chapter 4. Table H.1 repeats the error evidence available from the metrics and grids.

Table H.1: Error evidence available from the metrics and prediction grids

Model	Typical error type from metrics	Image level evidence available
1	balanced; small boundary differences	difference maps (Figure 4.15)
2	slight recall > precision; large outlier in HD95 (268 px)	difference maps (Figure 4.16)
3	wide spread; some very poor images (min Dice 0.123)	prediction grid only
4	over segmentation; at least one fully missed wound	prediction grid only

Appendix I: Hypergraph Visualisations

These figures are outputs of the notebooks. Figure A.1 is the schematic produced for Model 1 and Figure A.2 shows the nodes and dynamic hyperedges of Model 2 on a wound image.

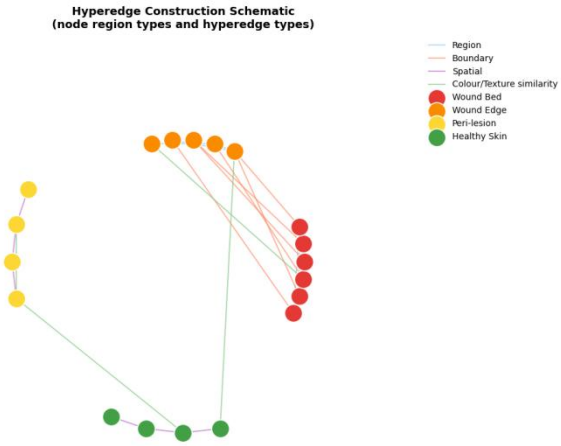


Figure A.1: Model 1 hyperedge construction schematic (notebook output)

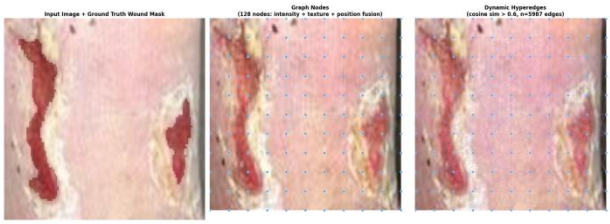


Figure A.2: Model 2 input image with reference mask, graph nodes and dynamic hyperedges (notebook output)

Appendix J: Symbolic Fusion Visualisations

Figure A.3 shows the Model 3 nodes by region type with the fixed symbolic hyperedges, and Figure A.4 the symbolic and visual hyperedges of one sample side by side.

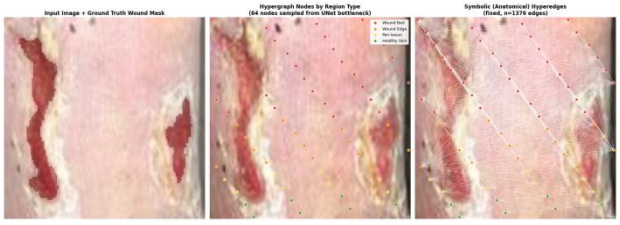


Figure A.3: Model 3 hypergraph nodes by region type and symbolic hyperedges (notebook output)

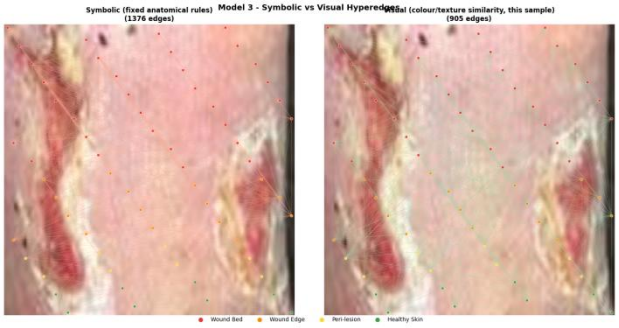


Figure A.4: Model 3 symbolic and visual hyperedges for one sample (notebook output)

Appendix K: Multi Granularity Analysis

Figure A.5 shows the fine, mid and coarse node positions of Model 4 and its cross scale hyperedges on a wound image.

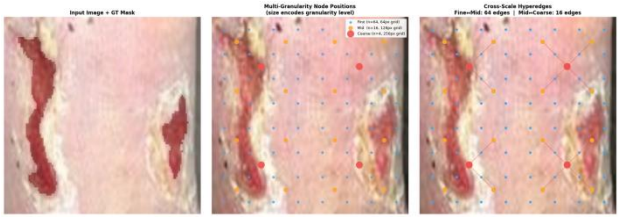


Figure A.5: Model 4 multi granularity node positions and cross scale hyperedges (notebook output)

Appendix L: Computational Performance

Table L.1: Parameter counts by module (millions)

Module	**Model 1**	**Model 2**	**Model 3**	**Model 4**
Encoder	23.508 (ResNet 50)	87.564 (ConvNeXt B)	7.826 (U Net)	23.749 (stem + residual + Mamba stages)
Vision Mamba bottleneck	29.604	7.462	1.896	included in encoder
Node construction	0.550	0.268	0.279	0.196
Hypergraph reasoning	1.909	0.988	0.132 (fusion gate)	1.977
Symbolic / fusion module	0.131 + 0.919	none	included above	0.657
Decoder	5.108	3.290	3.164	9.173
Total	**61.729**	**99.573**	**13.296**	**35.753**

Table L.2: FLOPs for one 512 × 512 input

Model	**GFLOPs**
1	91.15
2	not measured
3	not measured (input would be 384 × 384)
4	111.56

Table L.3: Inference latency in the experimental environment (single image, no TTA)

Model	Latency (ms)	Throughput (images/s)	Six fold TTA, estimated (ms)
1	47.03	21.26	282.16
2	130.29	7.68	781.7 (derived: 6 × 130.29)
3	103.98	9.62	623.9 (derived: 6 × 103.98)
4	40.55	24.66	243.32

Appendix M: Implementation Environment

The four notebooks were run on Kaggle. Table M.1 gives the hardware and software environment as recorded or reported.

Table M.1: Hardware and software environment

Item	Value
Platform	Kaggle notebooks
GPU	Two NVIDIA T4 GPUs (as stated by the authors); the code selects the single device cuda and uses no multi GPU wrapper
Framework	PyTorch 2.10.0+cu128
Other library versions (torchvision, NumPy, SciPy, scikit learn, OpenCV, thop)	Not recorded
Training time per model	Not recorded
Mixed precision	not used (no autocast in the code)

Appendix N: Source Code

The complete source code of the four models, with the training and evaluation notebooks, is available in the project repository:

https://github.com/Haroon-89/wound-seg