Practical Semantic Segmentation with Deep Learning

Training Models on Real-World Data

Book 2 | Practical Computer Vision with Deep Learning

Dr. Ali Khan | **Dr. Somaiya Khan**

March 2026

Chapter 1

From Benchmarks to Real Medical Imaging Problems

Benchmark datasets exist to provide a shared, reproducible measure of model performance. They serve an important purpose in research: they allow different models to be compared fairly under identical conditions. But they are designed for comparison, not for deployment. The conditions under which a benchmark model is trained and evaluated are rarely the conditions a model encounters in practice.

This gap is especially pronounced in medical imaging. A model trained on the ISIC skin lesion benchmark may encounter dermoscopy images from a different camera, with different calibration, acquired at a different institution, from a patient population with different demographic characteristics. The benchmark number tells you how the model performs under controlled conditions. It does not tell you how it will perform under yours.

1.1 The Benchmark Performance Gap

The gap between benchmark performance and deployment performance has been documented across medical imaging tasks. Models that achieve near-human performance on benchmark datasets under standard evaluation conditions frequently show substantially degraded performance when applied to data collected in different clinical settings.

This degradation has several causes that are specific to medical imaging and that do not appear with the same severity in general computer vision datasets.

Acquisition variability. Medical images are produced by equipment that varies across manufacturers, settings, and maintenance states. Dermoscopy images differ by device model. Unlike natural photographs, where cameras vary but the underlying scene statistics are relatively stable, medical imaging equipment variability introduces structured distribution shifts that can substantially affect model predictions.

Annotation subjectivity. The boundaries of a skin lesion, the extent of a polyp, or the margin of a tumor are not always unambiguous. Different clinicians annotating the same image may produce different masks. Benchmark datasets typically use one annotator per image, or adjudicated consensus, but the disagreement between expert annotators on ambiguous cases is rarely reported. Models trained on such data inherit this subjectivity.

Clinical selection bias. Images that appear in benchmark datasets are not always representative of the full clinical distribution. ISIC contains dermoscopy images from academic dermatology centers with relatively high-quality imaging protocols. The model learns from the cases that were selected, not from the full spectrum it will encounter in deployment.

1.2 What This Book Addresses

This book does not solve the generalization problem; no single book can. What it does is equip readers with the understanding and practical tools to reduce the gap between benchmark performance and real-world performance for medical segmentation tasks.

The core argument of this book is this: *most of the difference between a model that works on the benchmark and one that works in the clinic comes from decisions made before training begins.*

How the problem is defined, how data is collected, how annotations are created and validated, how class imbalance is addressed, and how the model is evaluated on clinically relevant subgroups; these decisions collectively determine model quality more than the choice of architecture or hyperparameters.

1.3 Dermoscopic Skin Lesion Segmentation as Case Study

This book develops its practical discussion around dermoscopic skin lesion segmentation because it captures many of the challenges that appear across medical segmentation tasks while remaining accessible for beginners and researchers entering the field.

Dermoscopic segmentation is a binary segmentation problem where the goal is to separate the lesion region from surrounding healthy skin. Although conceptually simple, the task contains many of the difficulties encountered in real medical imaging applications: irregular boundaries, large variation in lesion size, acquisition differences between devices, annotation ambiguity, and imaging artefacts such as hair, ruler markings, and bubbles.

Chapter 2

Problem Definition and Domain Analysis

Before writing a single line of training code, a medical segmentation project requires a clear and honest answer to a question that is easy to overlook: exactly what is the model supposed to segment, in what images, under what conditions, and for what clinical purpose?

Vague problem definitions produce vague annotations, which produce unreliable models. A model trained to segment "skin lesions in dermoscopy images" means something different depending on whether the definition includes seborrhoeic keratoses alongside melanomas, whether partial lesions at the image boundary are annotated, whether atypical benign lesions are included, and what the minimum lesion size for annotation is. Every one of these ambiguities propagates into the training data and affects what the model learns.

2.1 Defining the Segmentation Target

The first step in any medical segmentation project is to define the target structure precisely enough that two independent annotators would produce comparable masks. This definition is called the **annotation scope**.

An annotation scope document specifies:

- the anatomical or pathological structure to be segmented
- the imaging modality (dermoscopy, endoscopy, MRI, CT, etc.)
- how to handle ambiguous cases (e.g. lesions with unclear margins, polyps partially obscured by folds)
- the minimum size threshold below which structures are not annotated
- how to handle structures at the image boundary
- what constitutes background in this specific domain

For skin lesion segmentation, the scope must specify whether the segmentation boundary follows the dermoscopic edge of the lesion (which may be broader than the visible pigmentation) or the inner pigmented area only. These are two different segmentation tasks producing different masks from the same image.

2.2 Understanding the Clinical Context

A medical segmentation model is not an end in itself. It supports a clinical workflow, and understanding that workflow shapes every subsequent decision about data, training, and evaluation.

Diagnostic assistance. If the model assists a clinician in identifying and measuring a lesion, precision of boundaries matters greatly, but some false positives (over-segmentation) may be acceptable because the clinician can correct them. In this context, recall is more important than precision: missing part of the lesion is worse than including a small margin of healthy tissue.

Treatment planning. If segmentation outputs feed directly into a treatment planning system, the tolerance for error is much lower and precise boundaries are critical. Under-segmentation that leaves part of the lesion outside the segmented region can have direct clinical consequences.

Population screening. If the model is used to triage a large number of images and flag those requiring clinical attention, the cost of false negatives (missed findings) may be much higher than the cost of false positives (unnecessary follow-up). This asymmetry should be reflected in the choice of loss function and evaluation metric.

Understanding which of these contexts applies determines the correct trade-off between precision and recall, which in turn determines the appropriate loss function and threshold during inference.

2.3 Characterizing the Data Distribution

Before collecting or using a dataset, the practitioner should understand its distribution across the variables that matter for generalization.

Chapter 3

Medical Imaging Dataset Sources and Structure

Medical segmentation datasets differ from general computer vision datasets in ways that affect almost every aspect of the training pipeline. Images are produced by specialized equipment rather than cameras. Annotations require clinical expertise and are expensive to obtain. Dataset sizes are often one or two orders of magnitude smaller than general vision benchmarks, and privacy regulations restrict how data is shared and stored.

This chapter examines the dermoscopic skin lesion datasets used throughout this book, the ISIC family and the PH2 dataset, covering their characteristics, structure, common artefacts, and practical loading considerations.

3.1 The ISIC Skin Lesion Dataset Family

The International Skin Imaging Collaboration (ISIC) archive is the largest publicly available collection of dermoscopy images with expert annotations. Images are acquired using dermoscopes, devices that use polarized light and magnification to image skin lesions with reduced surface glare, and annotated by board-certified dermatologists.

The ISIC archive has been released in several versions through annual challenges. This book uses two versions: ISIC 2017, and 2018.

3.1.1 ISIC 2018

ISIC 2018 is the primary dataset used throughout this book, containing a total of 2,694 images. For consistency in the experiments presented in this book, a 70:30 split is used, where 1,886 images are allocated to training and 808 to testing.

- **Image resolution**: typically 700×900 pixels, resized to 256×256 for training
- **Annotation**: binary masks delineating the lesion boundary, produced by expert dermatologists
- **Classes**: two (lesion and background/healthy skin)
- **Lesion types**: melanoma, melanocytic naevi, seborrhoeic keratosis, basal cell carcinoma, and others

3.1.2 ISIC 2017

ISIC 2017 comprises 2,150 images, with 1,500 allocated to training and 650 to testing. It follows the same binary segmentation protocol as ISIC 2018 but was collected in an earlier period and exhibits a different distribution of lesion types and camera devices.

3.1.3 HAM10000

The Human Against Machine with 10000 Training Images (HAM10000) dataset is the largest dermoscopic dataset in this book, containing 10,015 images from multiple dermatoscopic sources including different hospitals, clinics, and acquisition devices. With the 70:30 split, 7,010 images are used for training and 3,005 for testing.

HAM10000's multi-source origin makes it a more challenging and more realistic dataset than either ISIC version. Models trained on it must generalize across acquisition device differences, which more closely reflects the clinical deployment scenario.

3.2 The PH2 Dataset

PH2 is a small dermoscopic dataset produced at the Dermatology Service of Hospital Pedro Hispano in Portugal, containing 200 images: 140 for training and 60 for testing. All images were acquired under the same controlled conditions using a Tuebinger Mole Analyzer system at 20× magnification.

PH2 serves as a controlled testbed: its small size and single-site acquisition isolate device and protocol variability. Strong performance on HAM10000 combined with strong performance on PH2 provides evidence that a model generalizes across both scale and device diversity.

Figure 3.1 presents representative samples from the four dermoscopic datasets used in this book.

Chapter 4

Annotation Standards and Quality Control

Annotation quality is the most direct determinant of training data quality. A model can only learn what its training labels encode. If the labels are inconsistent, biased, or systematically wrong at boundaries, the model will learn those inconsistencies and reproduce them in its predictions.

In medical imaging, annotation quality is particularly difficult to ensure because the task requires clinical expertise, the correct answer is often genuinely ambiguous, and the cost of systematic annotation errors is not always apparent until the model is evaluated in a clinical setting.

4.1 Sources of Annotation Error

Annotation errors in medical segmentation fall into four categories.

Boundary placement error. The annotator places the boundary in the wrong location, typically too tight (under-annotation) or too loose (over-annotation). In skin lesion segmentation, under- annotation is more common, with annotators drawing the boundary inside the visible dermoscopic margin. Over-annotation occurs when the annotator includes perilesional inflammation or reactive changes around the lesion boundary.

Missing regions. Parts of the target structure are omitted from the annotation. This is most common for small satellite lesions adjacent to the main lesion. Missing regions create false negatives in the training signal.

Inclusion errors. Non-target structures are included in the mask. Hair shadows and ruler marks are sometimes included in lesion masks when the annotator is unsure of the true boundary. These inclusions teach the model to segment the artefact alongside the lesion.

Inconsistent scope. Different annotators apply the annotation scope differently. One annotator may include perilesional changes; another may not. This inter-annotator inconsistency cannot be corrected through model training, it must be resolved at the annotation stage.

4.2 Measuring Inter-Annotator Agreement

Before using a dataset for training, measuring the agreement between independent annotators on the same images provides a realistic upper bound on model performance. A model cannot reliably outperform the human annotators whose labels it is trained on.

Inter-annotator agreement for segmentation is typically measured as the average Dice coefficient between pairs of annotations produced independently by different experts:

$$\text{Agreement} = \frac{1}{N}\sum_{n=1}^{N}\frac{2|A_n^{(1)} \cap A_n^{(2)}|}{|A_n^{(1)}| + |A_n^{(2)}|} \tag{4.1}$$

where $A_n^{(1)}$ and $A_n^{(2)}$ are the masks produced by two annotators for the n-th image.

For ISIC skin lesion segmentation, published inter-annotator Dice scores between expert dermatologists typically fall in the range 0.75–0.85, depending on lesion type. A model with validation Dice above the inter-annotator agreement is generally considered to have reached human-level performance on that task.

4.3 Annotation Validation Workflow

For projects that generate new annotations rather than using existing benchmark labels, a structured validation workflow reduces systematic errors.

1. **Annotator training.** All annotators review the annotation scope document and annotate a set of example images with known ground truth. Agreement with the reference is measured before proceeding.
2. **Independent annotation.** Each image is annotated independently by at least two annotators.
3. **Agreement measurement.** Dice agreement between pairs of annotations is computed automatically. Images with agreement below a threshold (e.g. Dice < 0.70) are flagged for review.
4. **Adjudication.** Flagged images are reviewed by a senior clinician who produces a final adjudicated mask.

Chapter 5

Handling Class Imbalance in Medical Segmentation

Class imbalance is a fundamental characteristic of dermoscopic segmentation datasets rather than an edge case. In dermoscopic images, lesion regions often occupy substantially fewer pixels than healthy skin, meaning the background class dominates the training signal.

This imbalance is not a problem with the datasets themselves. It reflects clinical reality: healthy skin occupies far more pixels than lesion tissue. However, this creates a learning challenge: a model that predicts mostly background can achieve deceptively high pixel accuracy while failing to segment lesions correctly.

5.1 Why Imbalance Causes Problems

During training with unweighted cross-entropy, the gradient signal at each step is proportional to the number of pixels belonging to each class. For an image where the lesion occupies 10% of pixels and the background occupies 90%, the background contributes approximately nine times more gradient to each weight update.

The model therefore learns quickly to suppress lesion predictions because reducing background error decreases the overall loss more efficiently than improving lesion segmentation. This is not a bug in the optimizer, it is the optimizer doing exactly what it was asked to do. The loss function is not properly encoding the task importance.

5.2 Strategies for Managing Imbalance

5.2.1 Dice Loss

As introduced in Book 1, Dice loss is inherently balanced because it measures region overlap independently for each class, normalized by class size. In a binary segmentation task:

$$\mathcal{L}_{Dice} = 1 - \frac{2\sum_{i,j} p(i,j) \cdot g(i,j) + \epsilon}{\sum_{i,j} p(i,j) + \sum_{i,j} g(i,j) + \epsilon} \tag{5.1}$$

For medical binary segmentation, the combination of binary cross-entropy and Dice loss is the standard choice:

$$\mathcal{L} = \mathcal{L}_{BCE} + \mathcal{L}_{Dice} \tag{5.2}$$

5.2.2 Weighted Binary Cross-Entropy

An alternative approach assigns higher weight to the foreground (lesion) class in the cross-entropy calculation. The weight is typically set as the inverse foreground frequency:

$$w_{fg} = \frac{N_{total}}{2 \cdot N_{fg}}, \quad w_{bg} = \frac{N_{total}}{2 \cdot N_{bg}} \tag{5.3}$$

where N_{fg} and N_{bg} are the total foreground and background pixel counts across the training set.

Listing 5.1: *Computing class weights for weighted cross-entropy from the training set.*

```
import numpy as np
from PIL import Image
import os

mask_dir = 'masks/train'
total_pixels = 0
fg_pixels    = 0

for fname in os.listdir(mask_dir):
    mask = np.array(
        Image.open(os.path.join(mask_dir, fname)).convert('L')
    )
    binarised = (mask > 127)
    total_pixels += binarised.size
    fg_pixels    += binarised.sum()
```

Chapter 6

Transfer Learning for Medical Segmentation

Medical imaging datasets are small by the standards of general computer vision. ISIC 2018 contains approximately 2,000 training images. Training a deep encoder-decoder network from random weight initialization on datasets of this size almost always produces inferior results compared to starting from a pretrained encoder.

Transfer learning, initializing part or all of the network with weights learned on a larger dataset, is not merely a performance optimization in medical segmentation. For small datasets, it is often the difference between a model that learns meaningful features and one that fails to converge usefully.

6.1 Why Pretrained Encoders Work for Medical Images

The intuition behind transfer learning for medical imaging is sometimes questioned: ImageNet contains photographs of everyday objects, not dermoscopy images. How can features learned from photographs of cats and cars be useful for segmenting skin lesions?

The answer lies in what the early and middle layers of a deep network actually learn. The first few layers detect edges and color gradients, patterns that are universal across image types. Middle layers detect textures, shapes, and part-level patterns. Only the deepest layers develop representations that are strongly category-specific to the training domain.

For medical segmentation, the early and middle layer features, edge detection, texture sensitivity, spatial structure, transfer almost perfectly from natural images. The decoder and output layers, which must interpret these features in the context of the specific segmentation task, are trained from scratch on the medical data.

This asymmetry motivates the standard transfer learning protocol for medical segmentation: initialize the encoder with ImageNet- pretrained weights and train the full network on the medical data.

6.2 The U-Net with Pretrained Encoder

The standard transfer learning setup for medical segmentation replaces the U-Net encoder (originally trained from scratch) with a pretrained backbone such as ResNet-34, ResNet-50, or EfficientNet.

The decoder and skip connection layers retain the U-Net structure but are initialized randomly and trained on the medical data.

Listing 6.1: *U-Net with pretrained ResNet-34 encoder implemented in PyTorch from scratch.*

```
import torch
import torch.nn as nn
import torchvision.models as models

class ConvBlock(nn.Module):
    """Two conv layers with BN and ReLU."""
    def __init__(self, in_ch, out_ch):
        super().__init__()
        self.block = nn.Sequential(
            nn.Conv2d(in_ch, out_ch, 3, padding=1, bias=False),
            nn.BatchNorm2d(out_ch),
            nn.ReLU(inplace=True),
            nn.Conv2d(out_ch, out_ch, 3, padding=1, bias=False),
            nn.BatchNorm2d(out_ch),
            nn.ReLU(inplace=True),
        )
    def forward(self, x):
        return self.block(x)

class DecoderBlock(nn.Module):
    """Upsample + concatenate skip + conv block."""
    def __init__(self, in_ch, skip_ch, out_ch):
        super().__init__()
        self.up   = nn.ConvTranspose2d(in_ch, out_ch, 2, stride=2)
        self.conv = ConvBlock(out_ch + skip_ch, out_ch)
```