Practical Semantic Segmentation with Deep Learning

Advanced Architecture Design and Experiments

Book 3 | Practical Computer Vision with Deep Learning

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

March 2026

Chapter 1

Foundations of Modern Segmentation Architectures

Semantic segmentation architectures have evolved substantially since the introduction of fully convolutional networks in 2015. The fundamental structure, encoder, bridge, decoder, has remained consistent, but every component has been redesigned multiple times to address specific limitations in computational efficiency, feature representation quality, and boundary precision.

Understanding this evolution is not merely historical. Each generation of design improvements identifies a specific limitation in the previous approach and proposes a targeted remedy. Learning to read the architecture literature in this way, as a sequence of problem identifications and proposed solutions. This is the core skill that allows a researcher to contribute new improvements rather than only applying existing ones.

Although the examples and experimental results throughout this book focus on dermoscopic image segmentation, the architectural principles discussed are broadly applicable to other semantic segmentation domains.

1.1 The Encoder-Decoder Pipeline

All modern semantic segmentation architectures share the same basic structure: an encoder that compresses the input image into a compact feature representation, a bridge or bottleneck that captures the most abstract semantic content, and a decoder that restores spatial resolution and produces the pixel-wise output.

This structure was established by U-Net (Ronneberger et al., 2015) and has not been superseded, it has only been refined. The variation across architectures lies not in whether to use this structure, but in how each component is implemented.

1.1.1 What the Encoder Must Do

The encoder has two competing objectives. It must reduce spatial resolution to build a sufficiently large receptive field for contextual understanding, and it must preserve the representational capacity needed to distinguish semantically similar but visually different structures (subtle lesion boundaries versus surrounding healthy skin, for example). These two objectives are in tension: aggressive downsampling builds receptive field but discards spatial detail.

Modern encoder design addresses this tension through several strategies: depthwise separable convolutions that reduce parameter count while maintaining representational capacity, residual connections that facilitate gradient flow through deeper encoders, and adaptive kernel sizes that capture features at multiple scales within a single encoder block.

1.1.2 What the Bridge Must Do

The bridge between encoder and decoder, often referred to as the bottleneck stage, must capture global contextual information from the smallest feature map in the network. At this point, the feature map has been reduced to its minimum spatial extent (typically $H/16 \times W/16$ for a four-stage encoder). Each position in the feature map has a receptive field that spans most of the input image.

The bridge is therefore the only point in the network where global scene context is naturally available. How this contextual information is extracted and represented at the bridge directly affects the quality of predictions for objects that require global scene understanding to segment correctly, such as large lesions with ambiguous local boundaries.

1.1.3 What the Decoder Must Do

The decoder must reconstruct spatial detail from the compressed bottleneck representation, guided by the skip connections from the encoder. The central challenge is that upsampling is inherently lossy: the information discarded by the encoder's downsampling cannot be fully recovered from the bottleneck alone. The decoder relies on skip connections to access the spatial detail that was present before downsampling.

How efficiently the decoder integrates bottleneck features with skip connection features, and how effectively it reconstructs the fine boundary detail needed for precise segmentation, varies significantly across architectures.

Chapter 2

Understanding the Encoder-Decoder Design Space

The encoder-decoder pipeline is a framework, not a fixed architecture. Every component within it has been implemented in multiple ways, each with different trade-offs between accuracy, efficiency, and deployment suitability. Understanding the design space of each component is the prerequisite for making principled architectural improvements.

2.1 Encoder Design Variants

The encoder is the most varied component across segmentation architectures. Three broad families of encoder design have been explored:

Standard convolutional encoders. The original U-Net uses paired 3×3 convolutional blocks at each stage. This design is simple, effective, and well-understood, but computationally expensive due to the large number of parameters in the 3×3 kernels applied at every channel.

Residual encoders. ResNet-style encoders add identity shortcut connections that allow gradients to flow directly from output to earlier layers. This enables training of deeper networks without vanishing gradient problems and is the basis for most pretrained encoder backbones used in transfer learning.

Lightweight encoders. MobileNet-style encoders replace standard 3×3 convolutions with depthwise separable convolutions, reducing parameter count and GFLOPs substantially while attempting to preserve representational capacity.

Hybrid encoder blocks combine spatial feature extraction and channel refinement within a single module. Such designs typically integrate different convolution types, residual pathways, or feature refinement operations to improve representational efficiency while maintaining a lightweight computational footprint. RFRE, introduced later in this book, serves as one example of this design philosophy.

2.1.1 The Role of Kernel Size

Kernel size determines the spatial extent of feature detection at each encoder layer. A 3×3 kernel captures local texture and edge patterns across a 3×3 neighborhood. A 1×1 kernel operates purely on channel relationships without any spatial aggregation.

The combination of 3×3 and 1×1 kernels within a single encoder block is commonly used in modern segmentation architectures. The 3×3 convolution extracts spatial patterns and local structures, while the subsequent 1×1 convolution refines channel relationships without introducing additional spatial aggregation cost. This design is often more efficient than stacking multiple 3×3 convolutions while still providing sufficient representational capacity for feature extraction and refinement.

2.1.2 Residual Connections in Encoders

Residual connections add the encoder block's input directly to its output:

$$F_{out} = F_{conv}(x) + W_{shortcut} \cdot x \tag{2.1}$$

where $W_{shortcut}$ is typically a 1×1 convolution to match channel dimensions. This has two effects: it provides a gradient highway that allows backpropagation to reach earlier layers without vanishing, and it forces the convolutional path to learn a residual correction rather than the full mapping, which is an easier learning problem.

For lightweight encoders where the convolutional path has reduced capacity, residual connections are particularly important: they ensure that even if the lightweight path cannot capture all features, the shortcut preserves the input information intact.

2.2 Bridge Design Variants

The bridge between encoder and decoder has been approached through pooling-based and convolution-based strategies.

Spatial Pyramid Pooling (SPP) applies max-pooling at multiple kernel sizes to create a multi-scale feature representation. SPP is computationally efficient but tends to lose fine-grained spatial detail because max-pooling

Chapter 3

Identifying Limitations in Standard U-Net

The standard U-Net is an effective segmentation baseline. The experimental results in Book 2 confirm this: Dice scores of 0.88–0.94 across four dermoscopic datasets demonstrate that a well-configured U-Net remains a strong segmentation baseline. The same results also reveal several limitations that commonly appear in standard encoder-decoder segmentation architectures.

This chapter examines those limitations analytically and discusses how they motivate the development of more efficient segmentation modules and architectural refinements.

3.1 Encoder Limitations

3.1.1 Computational Redundancy in Double-Convolution Blocks

The standard U-Net encoder applies two consecutive 3×3 convolutional layers at each stage. Both layers use the same kernel size and operate on the same spatial resolution. This design is effective but introduces redundancy: the second 3×3 convolution applies repeated spatial aggregation at the same resolution, increasing computational cost while providing partially overlapping feature extraction.

The total parameter count of the U-Net encoder is dominated by these double-convolution blocks. For a four-stage encoder with channel counts 64, 128, 256, 512, the double-convolution blocks account for the majority of the 7.763M total parameters. Replacing one of the two 3×3 convolutions with a 1×1 convolution, which performs channel mixing without spatial aggregation, provides different and complementary information at lower computational cost.

3.1.2 Limited Fine-Grained Feature Capture

Standard 3×3 convolutions are effective at capturing local texture and edge patterns, but the uniform kernel size means the encoder applies the same spatial receptive field throughout. In dermoscopic segmentation, the target structures vary enormously in texture scale: the overall color gradient of the lesion is a large-scale feature, while the boundary irregularity and texture heterogeneity within the lesion are fine-grained features at a much smaller scale.

The repeated use of fixed 3×3 kernels throughout the encoder cannot optimally capture both scales. Many modern lightweight encoder designs address this issue by combining spatial feature extraction operations with channel refinement mechanisms, allowing complementary information to be captured within a single block.

3.1.3 Parameter Count and Deployment Cost

At 7.763M parameters and 13.746 GFLOPs, the standard U-Net requires substantial computational resources. On the NVIDIA RTX4090 used in this book, inference takes approximately 1.93 ms per image. Lower- resource hardware platforms would be expected to exhibit substantially higher latency and memory requirements.

Parameter count also affects training time, GPU memory consumption, and deployment flexibility. Reducing model size therefore broadens the range of hardware environments in which segmentation models can operate efficiently.

3.2 Bridge Limitations

3.2.1 Standard Pooling Loses Spatial Detail

The standard U-Net does not include a dedicated bridge module, it applies the same double-convolution block at the bottleneck as in the encoder stages. This means the bottleneck has no mechanism for explicit multi-scale contextual aggregation.

Some U-Net variants add SPP or ASPP at the bottleneck. SPP captures multi-scale context through max-pooling at different kernel sizes, but max-pooling is a non-differentiable, non-learnable operation that discards spatial precision. ASPP adds dilated convolutions at dilation rates 6, 12, and 18, but the large dilation rates produce sparse sampling patterns that can miss fine-grained features and introduce gridding artefacts.

Chapter 4

The Residual Feature Refinement Encoder (RFRE)

Modern lightweight segmentation architectures often seek to reduce the computational redundancy of traditional encoder blocks while preserving feature extraction capability. One common strategy is to combine spatial feature extraction with lightweight channel refinement inside a residual framework.

The Residual Feature Refinement Encoder (RFRE) serves as a practical case study illustrating how lightweight encoder design principles can be translated into a complete architectural module.

4.1 Motivation

The standard U-Net encoder applies two consecutive 3×3 convolutions at each stage. Both perform the same type of spatial aggregation over the same receptive field, adding computational cost without qualitatively extending the feature types captured.

The RFRE addresses this by pairing a 3×3 convolution (spatial detail) with a 1×1 convolution (channel mixing and dimensionality refinement). The two operations are complementary: the 3×3 kernel captures local texture patterns across a neighborhood, while the 1×1 kernel learns non-linear channel combinations without spatial aggregation. A residual shortcut ensures gradient flow and feature preservation across the block.

4.2 Architecture

Figure 4.1 illustrates the architecture of the Residual Feature Refinement Encoder (RFRE), which combines spatial feature extraction, channel refinement, and residual learning within a lightweight encoder block.

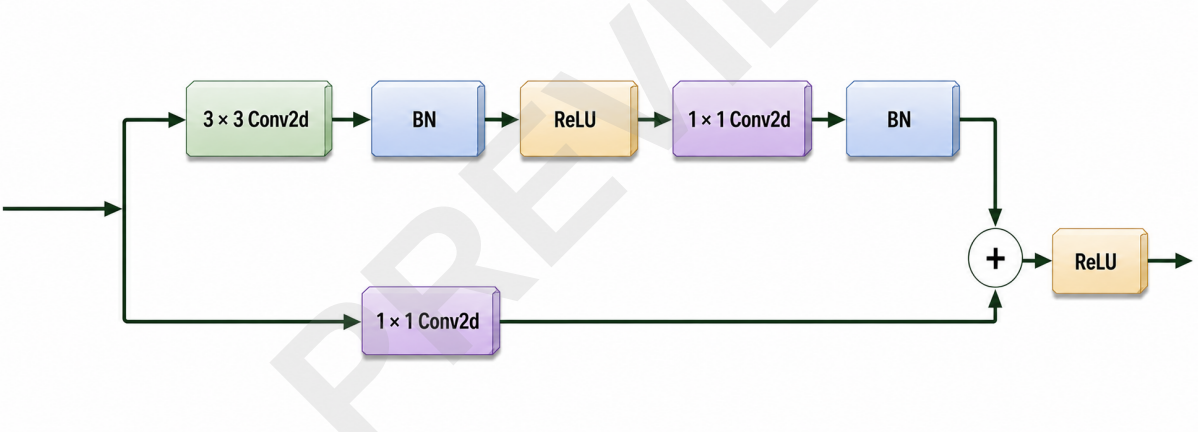

Figure 4.1: *Architecture of Residual Feature Refinement Encoder (RFRE).*

The RFRE block processes an input feature map $x \in \mathbb{R}^{C_{in} \times H \times W}$ and produces an output feature map $F_{out} \in \mathbb{R}^{C_{out} \times H \times W}$.

The main processing pathway consists of two consecutive convolutional operations:

1. A 3×3 convolution that extracts local spatial features such as edges, textures, and boundary patterns, followed by Batch Normalization and ReLU activation.
2. A 1×1 convolution that refines channel relationships and performs feature mixing across channels, followed by Batch Normalization.

In parallel, a residual shortcut projects the input feature map to the desired output dimensionality using a 1×1 convolution. This shortcut preserves information from earlier layers and facilitates stable gradient propagation during training.

The output of the main pathway and the shortcut pathway are then combined through residual addition and passed through a final ReLU activation:

$$F_{out} = \text{ReLU}\left(F_{conv}(x) + W_{sc} \cdot x\right) \tag{4.1}$$

Chapter 5

The Parallel Dilated Contextual Pyramid (PDCP)

Modern segmentation architectures often rely on specialized bridge modules to capture contextual information between the encoder and decoder. One common design strategy is to aggregate information at multiple receptive field scales so that both local details and global scene context can contribute to the final segmentation prediction.

The Parallel Dilated Contextual Pyramid (PDCP) serves as a practical example of this design philosophy.

5.1 Motivation

At the bottleneck, each feature map position has a receptive field spanning most of the input image. The bridge module determines how richly global and multi-scale context is represented before being passed to the decoder.

SPP captures multi-scale context through max-pooling but discards spatial precision. ASPP uses large dilation rates (6, 12, 18) that produce sparse sampling, missing fine-grained boundary details. Many modern bridge designs address these limitations by combining multiple contextual aggregation pathways, moderate receptive field expansion, and global context modeling. PDCP serves as one example of this design approach.

5.2 Design Rationale

The PDCP was designed around three primary objectives: capturing context at multiple scales, preserving dense feature sampling, and maintaining computational efficiency.

The first objective was to improve receptive field diversity. Objects and anatomical structures often contain information at multiple spatial scales. A bridge module should therefore be able to capture local detail, broader contextual information, and global scene characteristics simultaneously.

The second objective was to avoid the sparse sampling behavior that can occur when very large dilation rates are used. While larger receptive fields increase contextual coverage, excessively sparse sampling may reduce sensitivity to fine boundary details. Moderate receptive field expansion often provides a better balance between context and spatial precision.

The third objective was to maintain compatibility with lightweight encoder-decoder architectures. Bridge modules should enhance contextual representation without introducing excessive parameter growth or computational overhead.

Together, these objectives reflect a common principle in bridge design: increase contextual diversity while preserving efficiency and spatial fidelity.

5.3 Architectural Design

Figure 5.1 illustrates the architecture of the Parallel Dilated Contextual Pyramid (PDCP), which captures contextual information through multiple parallel pathways operating at different receptive field scales.

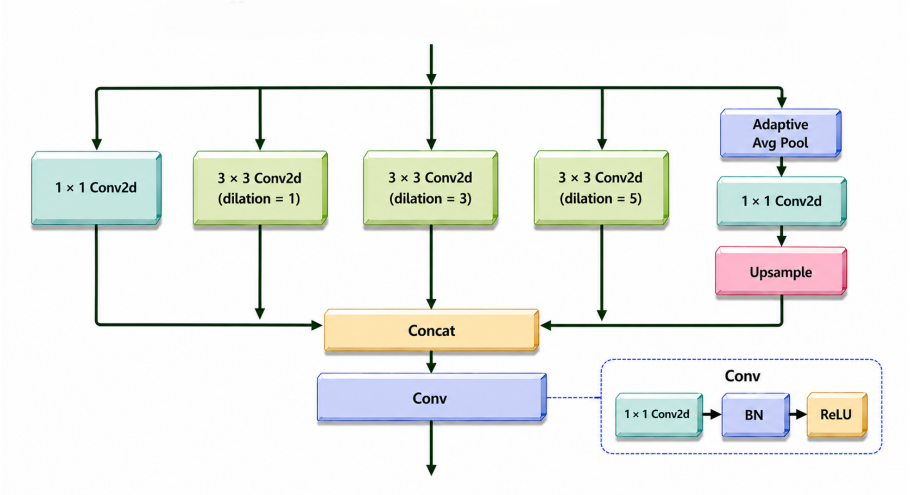

Figure 5.1: *Architecture of Parallel Dilated Contextual Pyramid (PDCP).*

Chapter 6

The Bottleneck Skip Fusion Decoder (BSFD)

Modern segmentation architectures often devote significant attention to decoder design because segmentation quality depends not only on extracting useful features but also on reconstructing precise spatial information.

One common design strategy is to combine efficient feature fusion, lightweight refinement, and artifact-free upsampling within the decoder pathway.

The Bottleneck Skip Fusion Decoder (BSFD) serves as a practical example of this design philosophy.

6.1 Motivation

The decoder is responsible for reconstructing high-resolution segmentation maps from compressed encoder representations. Although modern encoders are often the primary focus of architectural innovation, decoder design plays an equally important role in determining segmentation quality.

The standard U-Net decoder relies on transposed convolutions for upsampling and repeated convolutional blocks for feature refinement. While effective, these operations may introduce reconstruction artifacts and increase computational cost, particularly at high spatial resolutions.

Modern lightweight decoder designs therefore seek to improve feature fusion, reduce computational complexity, and preserve boundary information during reconstruction. The BSFD was developed as one example of this broader design direction.

6.2 Design Rationale

The BSFD was designed around three primary objectives: efficient feature fusion, artifact-free reconstruction, and lightweight spatial refinement.

The first objective was to improve the integration of encoder and decoder information. Skip connections provide detailed spatial information, while decoder features contain higher-level semantic information. Effective segmentation requires both.

The second objective was to avoid reconstruction artifacts. Upsampling operations should recover spatial resolution without introducing artificial patterns that may degrade segmentation quality.

The third objective was to maintain computational efficiency. Decoder stages often process high-resolution feature maps where computational cost can increase rapidly. Lightweight refinement mechanisms therefore become particularly valuable.

Together, these objectives reflect a common principle in decoder design: reconstruct detailed segmentation outputs while keeping computational requirements manageable.

6.3 BSFD Architecture

Figure 6.1 illustrates the architecture of the Bottleneck Skip Fusion Decoder (BSFD), which reconstructs high-resolution segmentation features through feature fusion, lightweight refinement, and spatial reconstruction.