Deep Learning & Generative AI in Healthcare

Session 06

Session Overview

Part I: Foundations & Medical Context

- Medical imaging modalities and clinical needs
- Problem formulation: Segmentation vs Classification
- Data characteristics and challenges

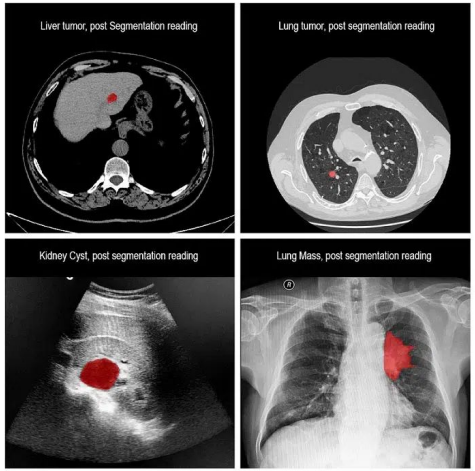
Part II: Segmentation Architectures

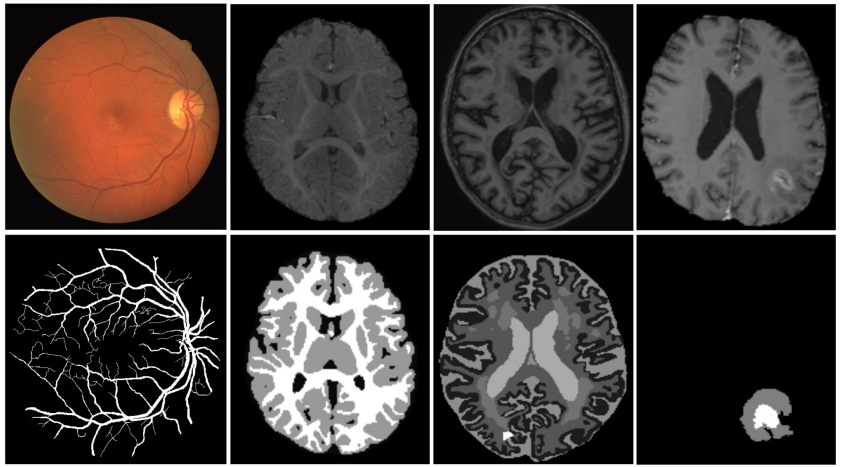
- U-Net family and encoder-decoder architectures
- Attention mechanisms and transformers
- Foundation models (SAM, MedSAM)

Session Overview

Part III: Classification & Diagnosis

- CNN architectures to Vision Transformers
- Multi-modal learning and foundation models
- Clinical deployment and evaluation





Medical Imaging: The Data Landscape

Radiology (2D/3D)

CT: High resolution, 3D volumes

MRI: Multi-sequence, soft tissue

X-ray: 2D projection, screening

Ultrasound: Real-time, operator-dependent

Pathology (2D)

Whole Slide Images: Gigapixel resolution

H&E Staining: Standard histology

Immunohistochemistry: Molecular markers

Multiplex imaging: Multi-channel data

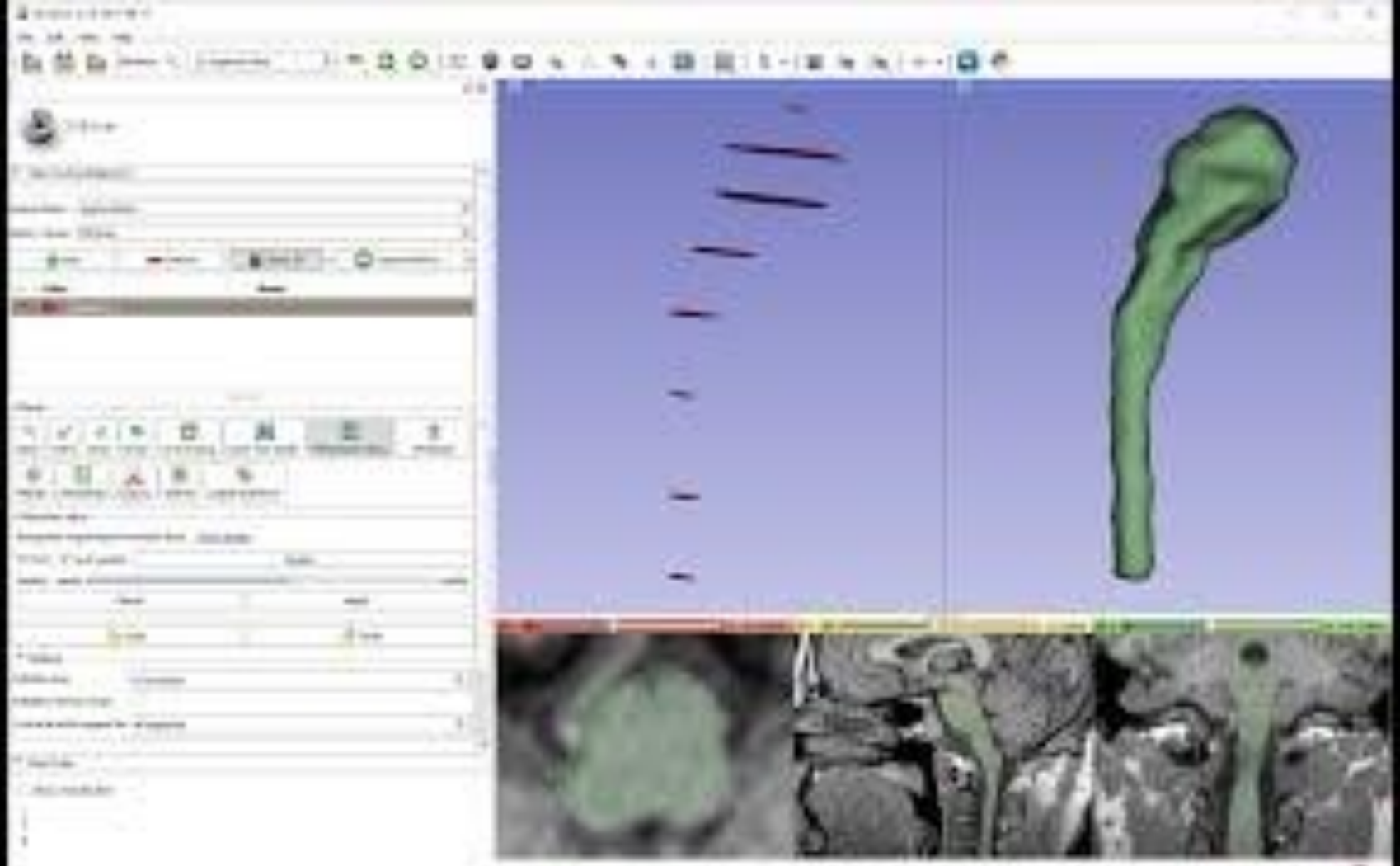
Key Characteristics

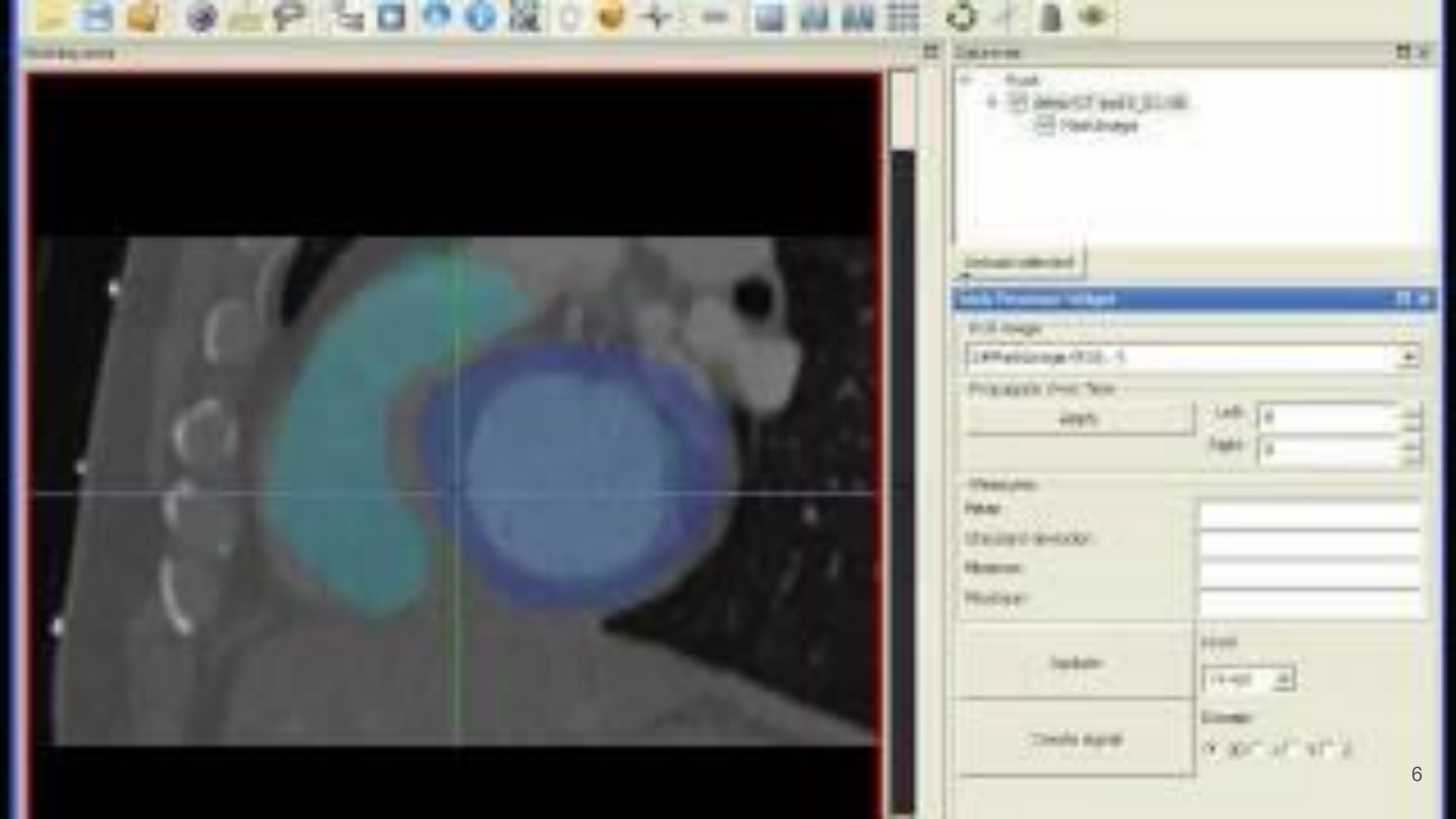
High dimensionality
512×512×300 voxels typical

Limited annotations
Expert time is expensive

Class imbalance
Pathology is rare

Critical Challenge: 90% of medical data is imaging, yet only ~1% is labeled for AI training





Segmentation vs Classification: Problem Formulation

Semantic Segmentation

Goal: Pixel/voxel-wise classification

Output: Dense prediction map (H×W×C)

Applications:

- Tumor delineation for radiotherapy
- Organ segmentation for volume quantification
- Lesion identification in CT/MRI
- Cell nuclei segmentation in pathology

```
Input: (B, C, H, W, D)
Output: (B, K, H, W, D)
```
K = number of classes

Image Classification

Goal: Image-level label prediction

Output: Class probabilities (C,)

Applications:

- Disease presence/absence detection
- Cancer subtype classification
- Severity grading (mild/moderate/severe)
- Quality control and artifact detection

```
Input: (B, C, H, W, D)
Output: (B, K)
```
K = number of classes

Clinical Insight: Segmentation provides interpretability and precise localization, while classification enables high-throughput screening. Modern systems often combine both.

Review: Convolutional Neural Networks

Core Operations

Convolution
Local feature extraction with learned kernels

```
Conv2D(in_ch, out_ch, kernel=3×3)
```

Pooling
Spatial downsampling (max/average)

```
MaxPool2D(kernel=2×2, stride=2)
```

Activation
Non-linearity (ReLU, GELU)

```
ReLU(x) = max(0, x)
```

Normalization
Batch/Instance/Group normalization

```
BatchNorm2D(num_features)
```

Key Properties

Translation Equivariance
If input shifts, output shifts accordingly

Local Receptive Fields
Each neuron sees limited spatial region

Parameter Sharing
Same kernel applied across spatial locations

Hierarchical Features
Early layers: edges → Later layers: objects

Medical Imaging Adaptation: 3D convolutions for volumetric data, smaller batch sizes due to memory, extensive data augmentation due to limited training data

U-Net: The Foundation of Image Segmentation

Ronneberger et al., 2015 (MICCAI) - 67,000+ citations

Designed specifically for biomedical image segmentation with limited training data

Architecture Components

Contracting Path (Encoder)

Repeated: Conv 3×3 → ReLU → Conv 3×3 → ReLU → MaxPool 2×2

Doubles channels at each downsampling

Bottleneck

Lowest spatial resolution, highest channel count

Captures semantic information

Expanding Path (Decoder)

UpConv 2×2 → Concatenate skip → Conv 3×3 → ReLU

Halves channels at each upsampling

Key Innovations

Skip Connections

Concatenate encoder features with decoder

→ Preserves spatial information lost in downsampling

Symmetric Design

Equal-sized encoder and decoder paths

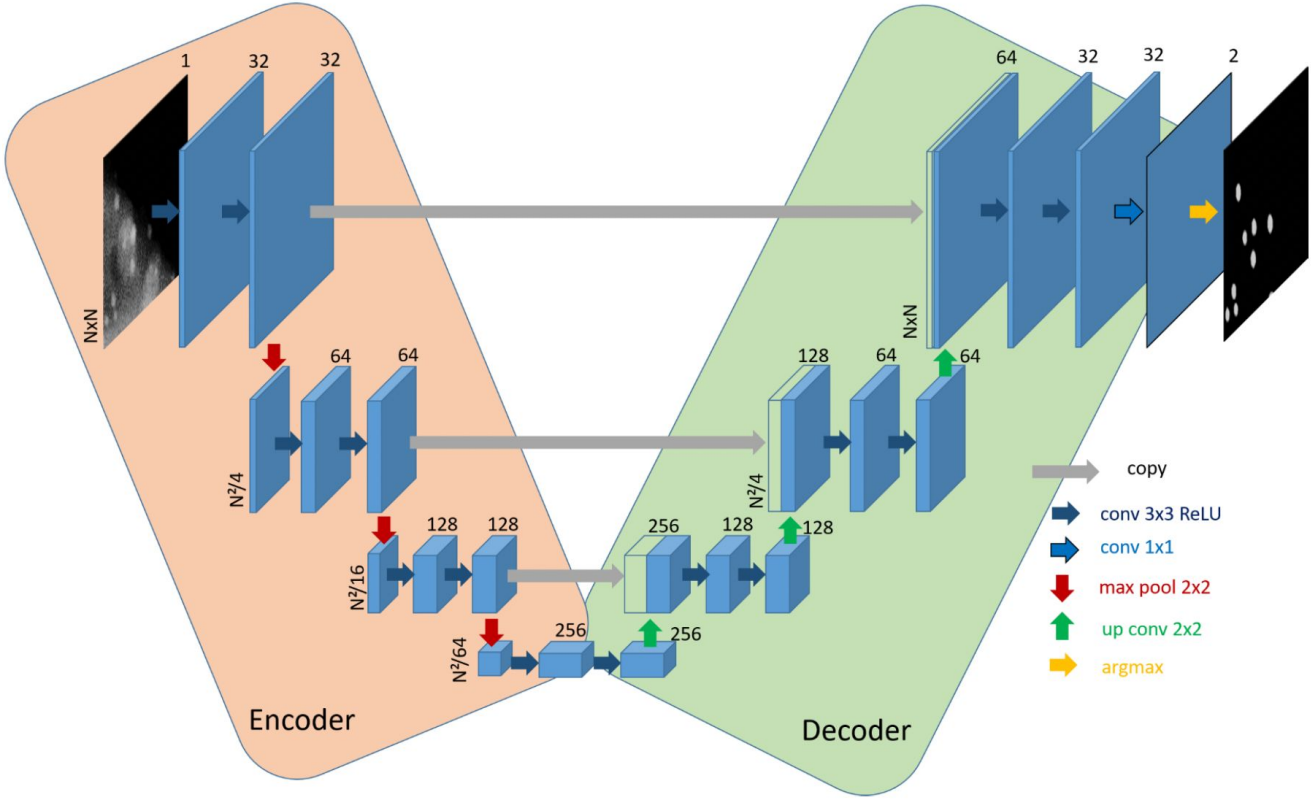
→ Balances context and localization

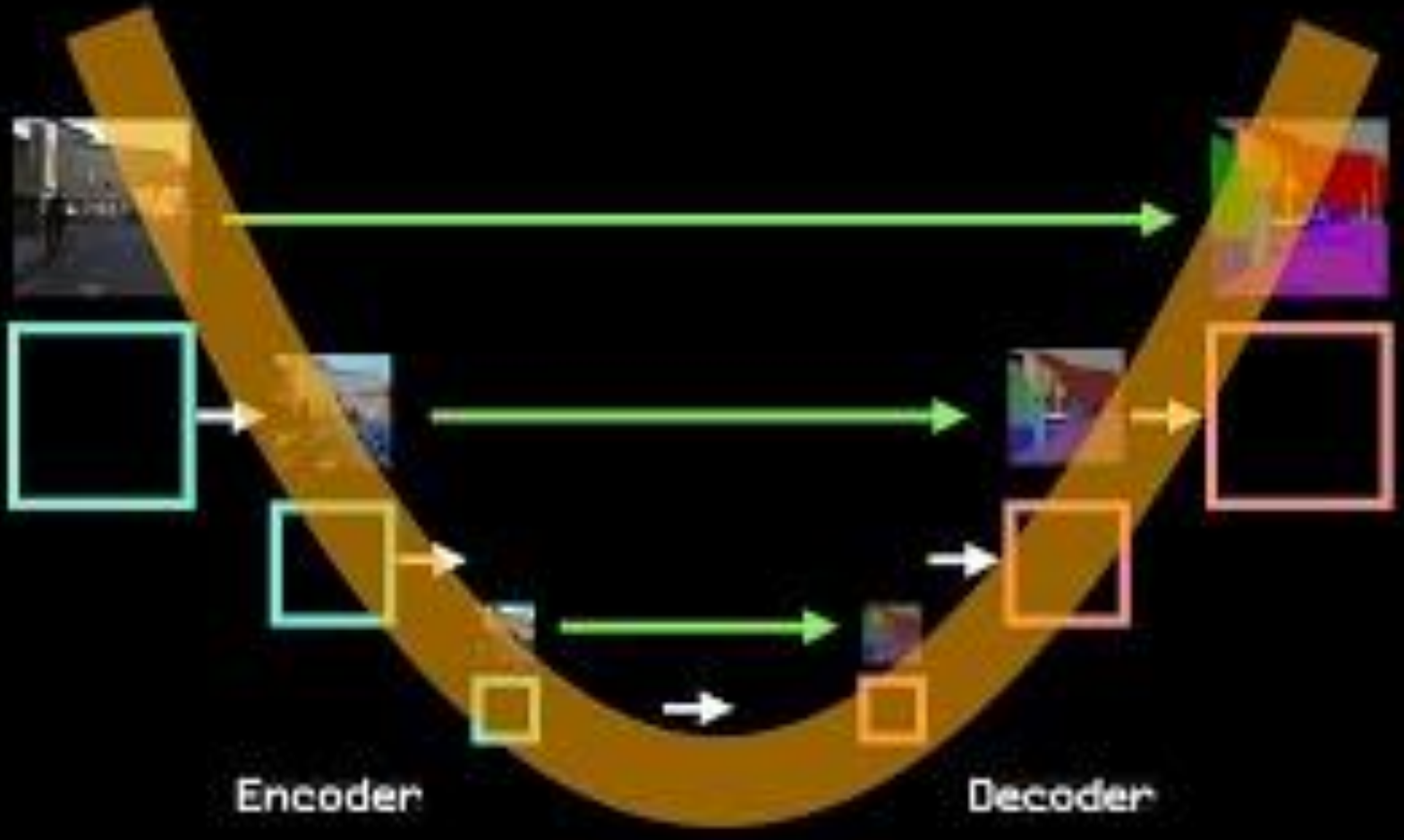
Elastic Deformations

Aggressive data augmentation strategy

→ Enables training with few annotated images

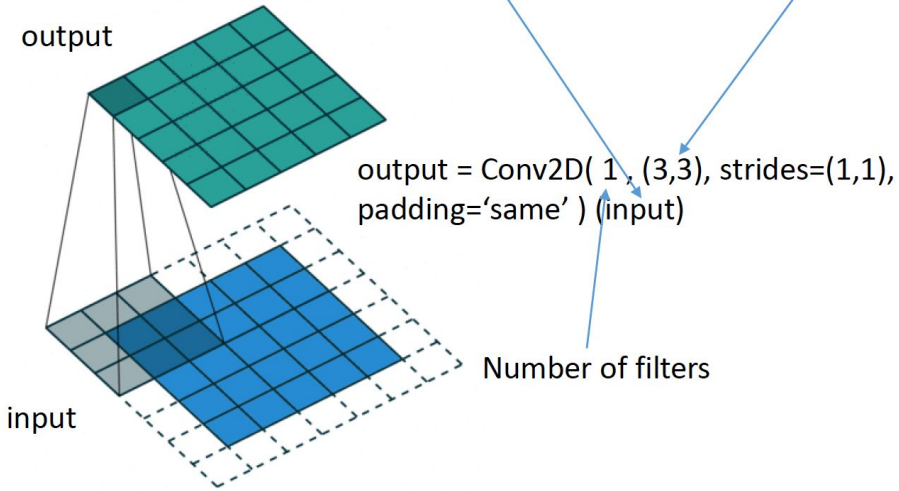
U-Net: The Foundation of Image Segmentation



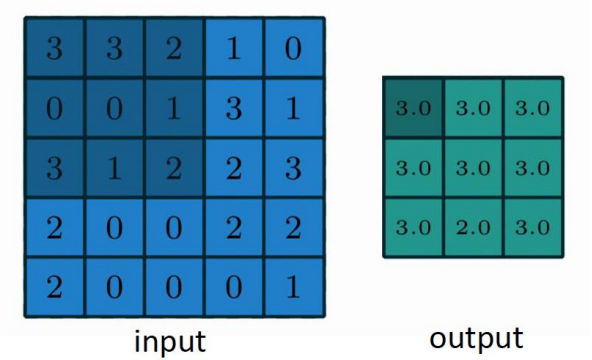

Encoder
Decoder

Convolutions + Max Pooling

2D convolution using a kernel size of 3, stride of 1 and padding

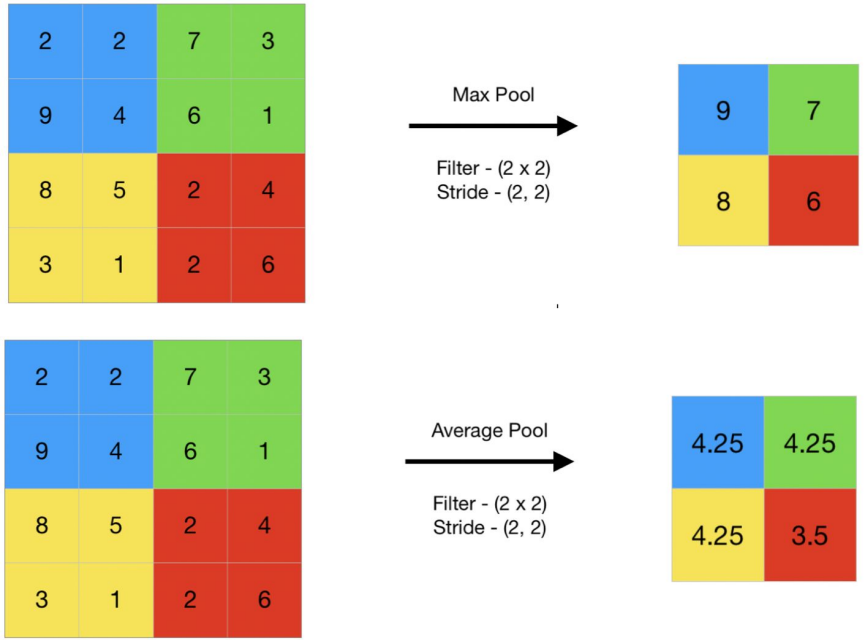


Max pooling kernel size of 3, stride of 1, no padding



output = MaxPooling2D((3, 3), strides=(1,1), padding='valid') (input)

Different pooling layers

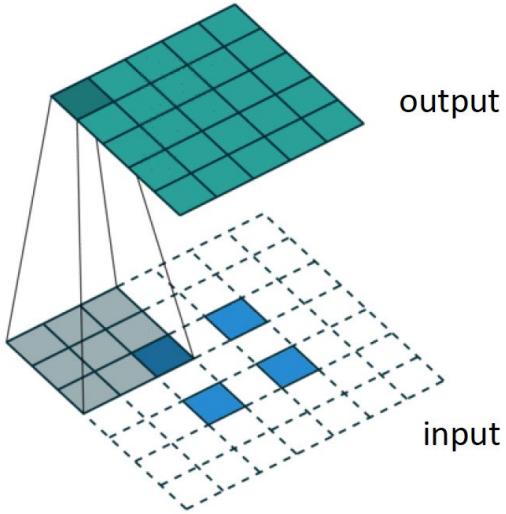


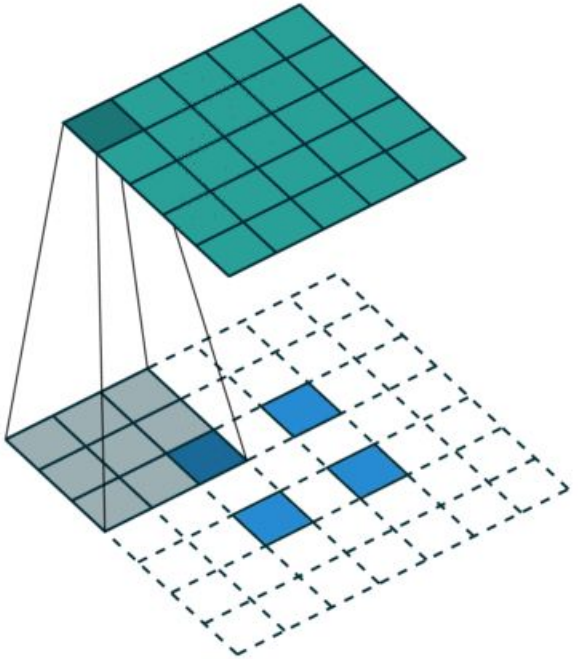

```python
import numpy as np
from keras.models import Sequential
from keras.layers import MaxPooling2D

# define input image
image = np.array([[2.0, 2.0, 7.0, 3.0],
                  [9.0, 4.0, 6.0, 1.0],
                  [8.0, 5.0, 2.0, 4.0],
                  [3.0, 1.0, 2.0, 6.0]])
image = image.reshape(1.0, 4.0, 4.0, 1.0)

# define model containing just a single max pooling layer
model = Sequential(
    [MaxPooling2D(pool_size = 2, strides = 2)])

# generate pooled output
output = model.predict(image)

# print output image
output = np.squeeze(output)
print(output)
```

https://www.geeksforgeeks.org/cnn-introduction-to-pooling-layer/

Transpose convolution - Upsampling

Transposed 2D convolution with padding, stride of 2 and kernel of 3

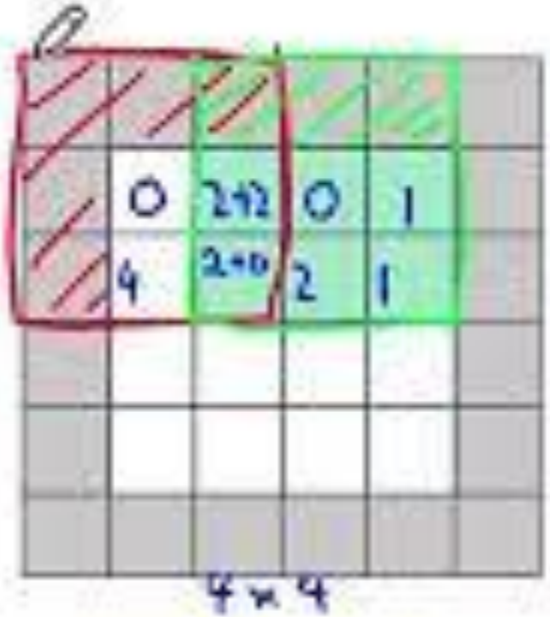




output = Conv2DTranspose(1, (3,3) , strides= (2,2), padding='same') (input)

Transpose Convolution

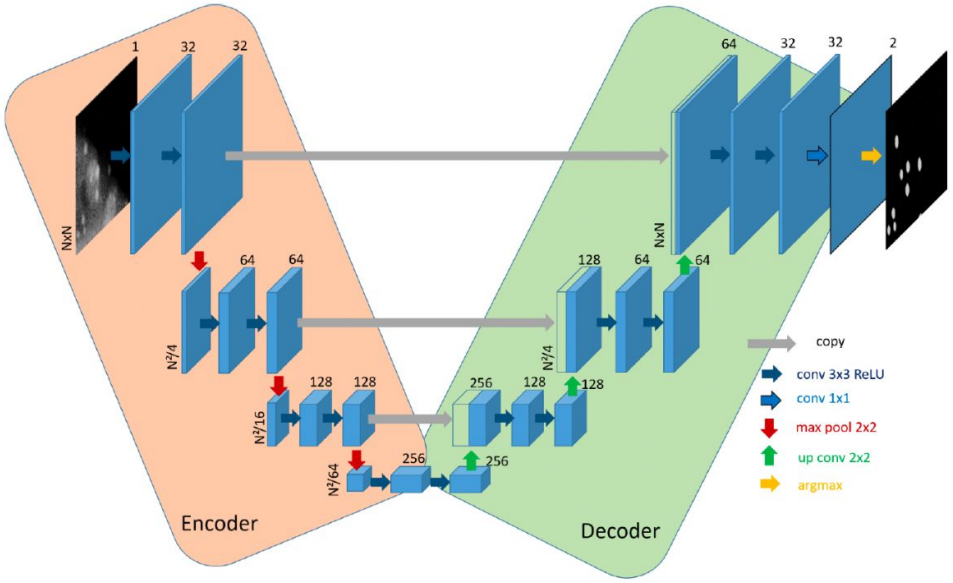
2×2

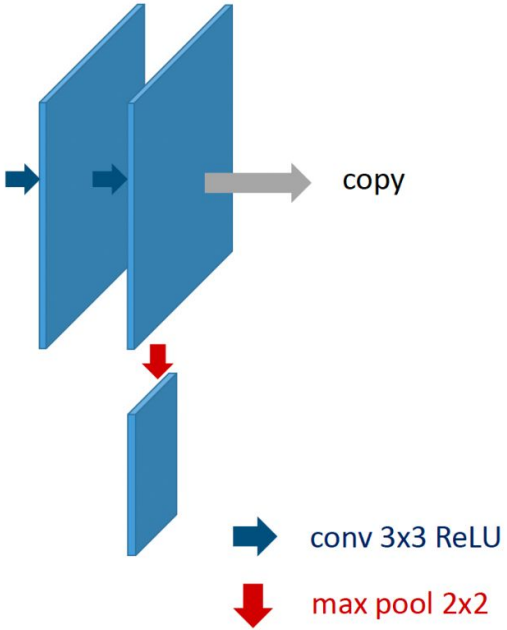


4×4

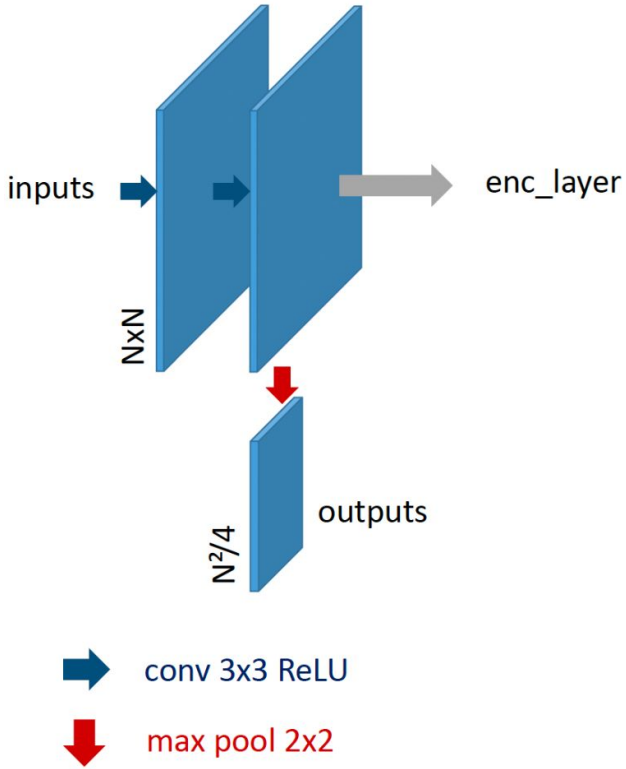
filter f×f = 3×3 padding p=1 stride s=2

UNet: Encoder



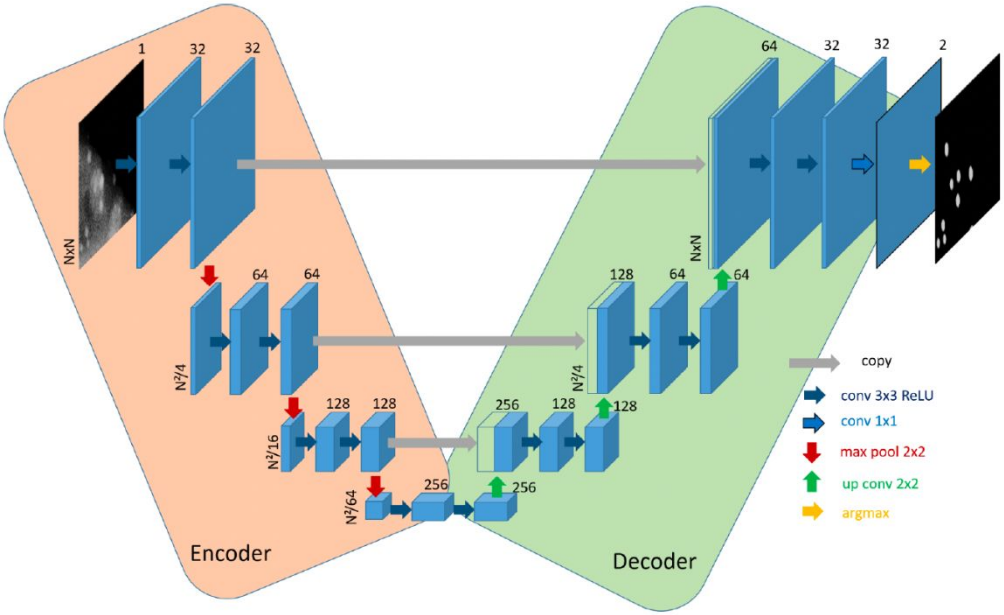


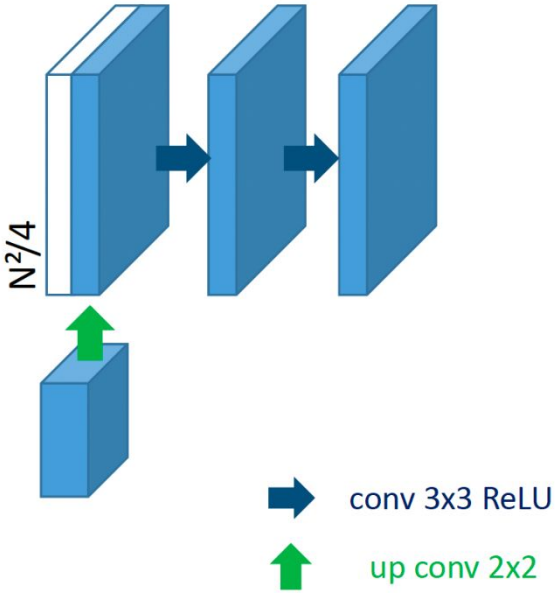
UNet: Encoder



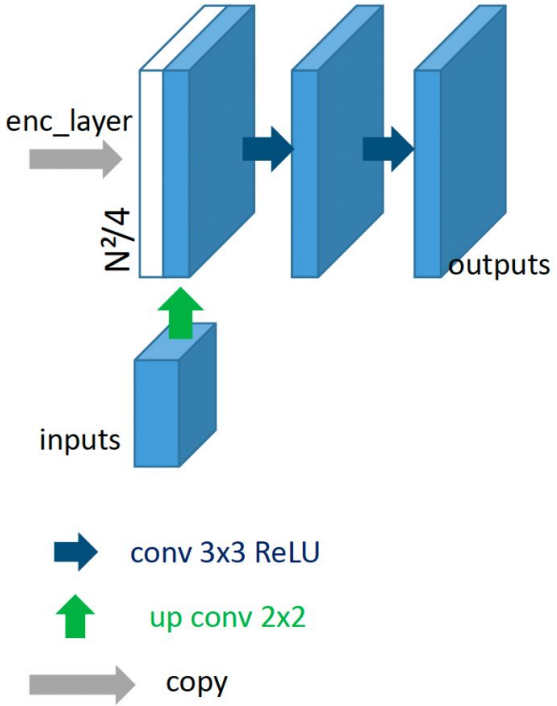

```
c = Conv2D(filters, (3,3), activation='relu',
kernel_initializer=kernel_initializer, padding='same') (inputs)

c = Conv2D(filters, (3,3), activation='relu',
kernel_initializer=kernel_initializer, padding='same') (c)

enc_layer = c

outputs= MaxPooling2D((2, 2)) (c)
```

UNet: Decoder



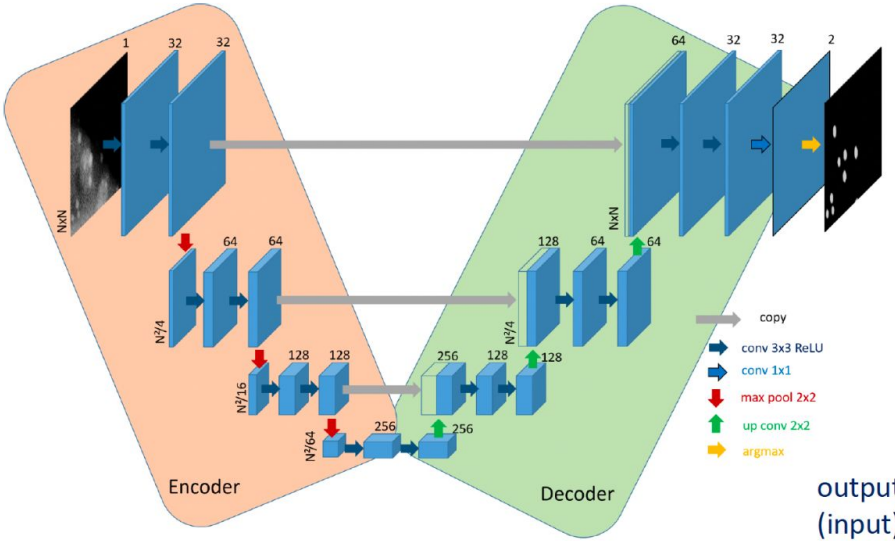


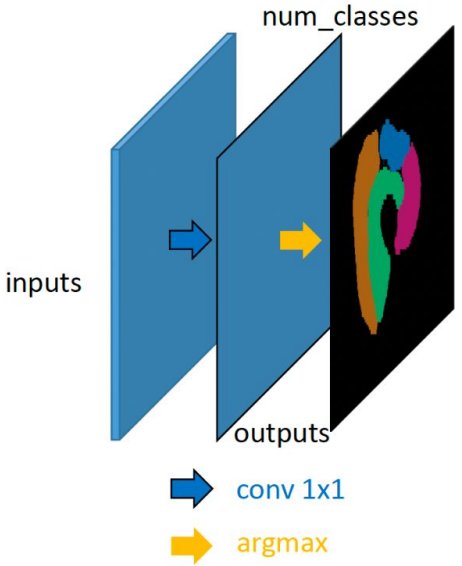
UNet: Decoder




```
c = Conv2DTranspose( filters, (2, 2), strides=(2, 2), padding='same') (input)

c = Concatenate()([c, enc_layer])

c = Conv2D(filters, (3,3), activation='relu', kernel_initializer=kernel_initializer,
padding='same') (inputs)

outputs = Conv2D(filters, (3,3), activation='relu',
kernel_initializer=kernel_initializer, padding='same') (c)
```

UNet: Decoder





outputs = Conv2D(num_classes, (1, 1), activation='sigmoid')
(input)

The argmax is done outside the network (loss or metric, display, ...)

U-Net Evolution: Modern Variants

3D U-Net (2016)

Extends to volumetric medical data

- 3D convolutions and 3D pooling
- Better for CT/MRI volumes
- Memory intensive: patch-based training

Attention U-Net (2018)

Attention gates in skip connections

- Learns to focus on relevant features
- Suppresses irrelevant regions
- Improved boundary delineation

U-Net++ (2018)

Dense skip pathways

- Nested and dense skip connections
- Reduces semantic gap between encoder-decoder
- Deep supervision at multiple scales

nnU-Net (2021)

Self-configuring framework

- Automated architecture adaptation
- Dataset-specific preprocessing
- State-of-art on Medical Segmentation Decathlon

nnU-Net: Key Principles

Automatic Preprocessing
Resampling, normalization based on data properties

Dynamic Architecture
2D, 3D, or cascade based on dataset

Robust Training
Optimized augmentation and loss functions

Attention Mechanisms & Vision Transformers

From Local to Global Context

CNNs have limited receptive fields. Transformers enable global context modeling through self-attention.

Self-Attention Mechanism

```
Attention(Q, K, V) = softmax(QK^T / √d_k)V
```

Query (Q): What am I looking for?

Key (K): What do I contain?

Value (V): What information do I have?

Advantage: Each position attends to all other positions, capturing long-range dependencies

Vision Transformer (ViT)

1. Patch Embedding
Split image into patches (16×16), linear projection

2. Position Encoding
Add learnable position embeddings

3. Transformer Encoder
Stack of self-attention + MLP blocks

4. Classification Head
MLP on [CLS] token or global pooling

Challenges in Medical Imaging

Data Hungry
ViT requires large datasets or pre-training

Computational Cost
$O(n^2)$ complexity for self-attention

Loss of Inductive Bias
No built-in translation equivariance

Hybrid Architectures: TransUNet & Swin-UNet

TransUNet (2021)

Combines CNN and Transformer for segmentation

Encoder
CNN backbone (ResNet) for low-level features

Bottleneck
Transformer encoder for global context

Decoder
Cascade upsampling with skip connections

Benefit: CNN captures local texture, Transformer captures global semantic relationships

Swin-UNet (2022)

Pure transformer with hierarchical design

Shifted Windows
Local attention within windows, then shifted

Patch Merging
Hierarchical feature maps (like CNN pooling)

Linear Complexity
O(n) instead of $O(n^2)$ for self-attention

Advantage: Efficient for high-resolution medical images, maintains hierarchical representation

Hybrid Architectures: TransUNet & Swin-UNet

Performance Comparison

Architecture	Synapse Multi-Organ	Parameters	Key Strength
U-Net	76.85% DSC	~31M	Simple, reliable
TransUNet	77.48% DSC	~105M	Global context
Swin-UNet	79.13% DSC	~27M	Efficiency

Foundation Models: Segment Anything (SAM)

Meta AI - April 2023

First foundation model for image segmentation trained on 1 billion masks

Architecture Overview

Image Encoder
ViT-H (Huge): 632M parameters
Pre-computable embeddings for efficiency

Prompt Encoder
Points, boxes, masks, or text
Flexible user interaction

Mask Decoder
Lightweight transformer
Predicts multiple mask proposals

Key Innovation: Promptable segmentation enables zero-shot transfer to new domains

Prompting Strategies

Point Prompts
Click on object of interest
Best for: Interactive annotation

Box Prompts
Bounding box around region
Best for: Object detection pipelines

Mask Prompts
Coarse mask refinement
Best for: Iterative segmentation

Automatic Mode
Grid of points for everything
Best for: Dense segmentation

MedSAM: Medical Adaptation of SAM

MedSAM (Nature Communications, 2024)

Fine-tuned SAM on 1.6M medical image-mask pairs across 11 modalities

Training Strategy

Dataset Composition

- CT: 760K images (tumors, organs)
- MRI: 480K images (brain, cardiac)
- Ultrasound: 150K images
- Microscopy: 210K images (cells, tissues)

Fine-tuning Approach

Freeze image encoder, train mask decoder and prompt encoder on medical data

Preserves natural image features, adapts to medical domain

Performance Improvements

Task	SAM	MedSAM
Liver CT	67.1%	**91.8%**
Brain MRI	48.2%	**84.3%**
Cell Nuclei	72.5%	**89.7%**

MedSAM: Medical Adaptation of SAM

Fine-tuning Approach

Freeze image encoder, train mask decoder and prompt encoder on medical data

Preserves natural image features, adapts to medical domain

Clinical Workflow Integration

Interactive Refinement

Radiologist provides box prompt → MedSAM segments → Add/remove points to refine

Quality Control

Rapid review of automated segmentations with minimal corrections needed

Training Data Creation

10x faster annotation for building task-specific models

Open Source: MedSAM models and code available at github.com/bowang-lab/MedSAM

Loss Functions for Segmentation

Cross-Entropy Loss

```
L = -Σ y_i log(p_i)
```

Pixel-wise classification loss

Pros:
- Well-studied, stable training
- Works with multi-class problems

Cons:
- Sensitive to class imbalance
- Treats pixels independently

Dice Loss

```
L = 1 - (2|X∩Y|)/(|X|+|Y|)
```

Based on Dice similarity coefficient

Pros:
- Handles class imbalance naturally
- Directly optimizes evaluation metric

Cons:
- Can be unstable for small objects
- Non-convex optimization surface

Focal Loss

```
L = -α(1-p)^γ log(p)
```

Down-weights easy examples

Use Case:
- Extreme class imbalance
- Small object detection
- Hard example mining

Boundary Loss

```
L = ∫ ϕ(s)ds
```

Penalizes boundary errors

Use Case:
- Precise boundary delineation
- Organ segmentation
- Tumor margin definition