Reimplementation of U-Net Paper

Sambhav Agarwal, Jonathan Levitsky, Sai Chamarty

ECS 174: Computer Vision

December 2025

Problem Statement

Problem 1

Biomedical Cell Segmentation is difficult. Microscopy images have overlapping cells, weak boundaries, and noise. Traditional image classification methods struggle at precise cell separation.

Problem 2

Original U-Net was limited by 2015 Training Methods. Original U-Net relied on outdated training (no BN, no ADAM, tiny datasets). Unclear how much improvement comes from *better optimization* rather than *new architectures*.

Problem 3

U-Net was evaluated mainly on small, homogeneous datasets. Modern datasets (e.g., DSB2018) have diverse cell types and different annotation styles, making consistent segmentation more difficult.

Big Question

Can we significantly improve U-Net performance *without changing the architecture*, simply by using modern training methods and larger datasets?

Method Overview

Datasets Used

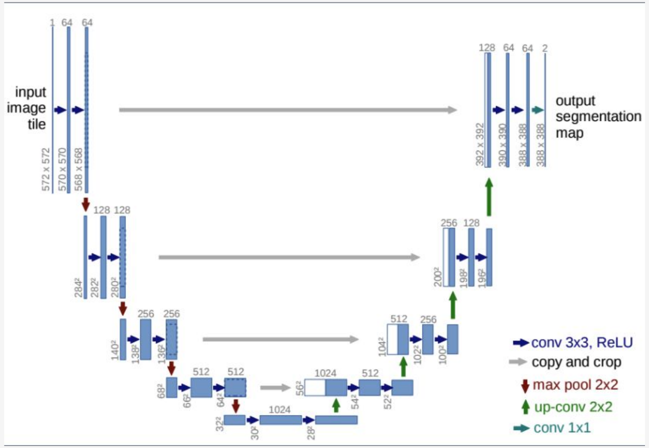
- **PhC-U373 (Cell Tracking Challenge)**: Small, high-quality phase-contrast microscopy dataset (~40 labeled images).
- **DSB2018 (Data Science Bowl)**: Large, diverse, instance-level nucleus segmentation dataset.

Data Augmentation (from paper)

- **Elastic Deformation**: Simulates realistic biological distortions; considered the key augmentation in the 2015 paper.
- **Random Flips & Crops**: Increase dataset variability and prevent overfitting on small microscopy datasets.
- **Normalization**: Standardized pixel intensities for more stable training.

Original U-Net Architecture (Reproduced in Pytorch)

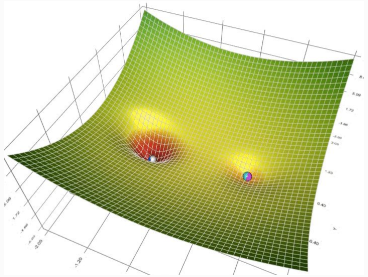
- Implemented **exact 2015 U-Net** structure manually in PyTorch (the paper used Caffe).
- **He initialization**, **valid convolutions**, and **copy-and-crop skip connections** exactly as described.
- Pixel-wise **boundary weighting** from the paper used to help the network separate touching cells.

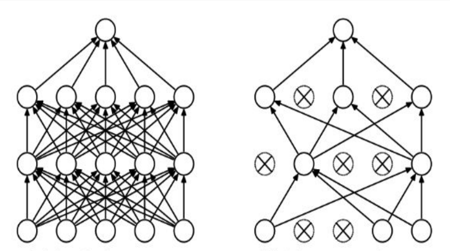


Modern Optimizations & Extended training

U-Net Updated with Modern Training Techniques

- **Batch Normalization**: stabilizes gradients and accelerates convergence.
- **ADAM optimizer**: replaces classical SGD from the original paper.
- **Dropout layers**: reduce overfitting on small datasets.
- **Learning-Rate Scheduler (ReduceLROnPlateau)**: adapts learning rate during plateaus.
- **Dice + Cross-Entropy Loss**: improves boundary precision and handles class imbalance.

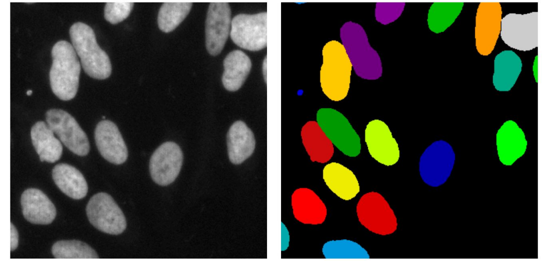




(a) Standard Neural Net (b) After applying dropout.

Training U-Net with Optimizers on DSB 2018

- Applied the modernized U-Net to the large, heterogeneous DSB2018 dataset.
- Converted instance masks to binary segmentation masks and generated instance-aware weight maps.
- Evaluated how well the optimized model scales to a broader distribution of cell types and imaging conditions.

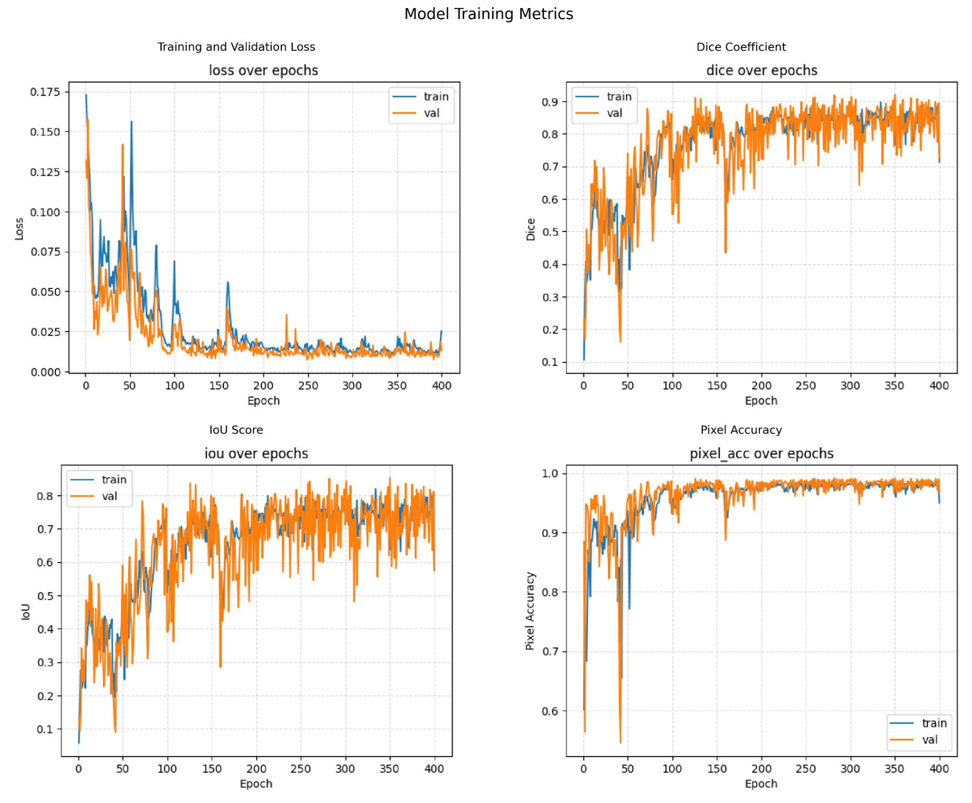


Experimental Results

01

Baseline Reproduction

- Dice: **91.95%**
- IoU: **85.21%**
- Highly noisy due to batch size = 1 and tiny dataset
- Results align closely with expectations from original paper reproduction

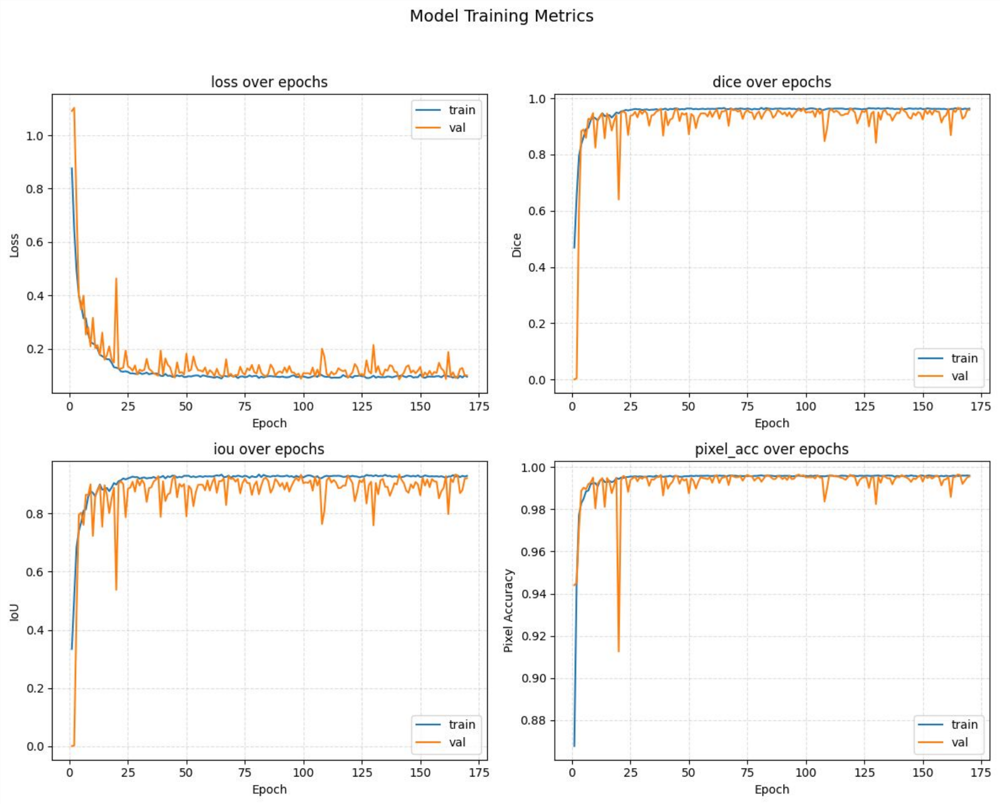


Experimental Results

02

With Modern Optimizers

- Dice improved to **96.2%**
- Faster and smoother convergence
- Better boundary separation and less overfitting
- IoU curves also improved significantly

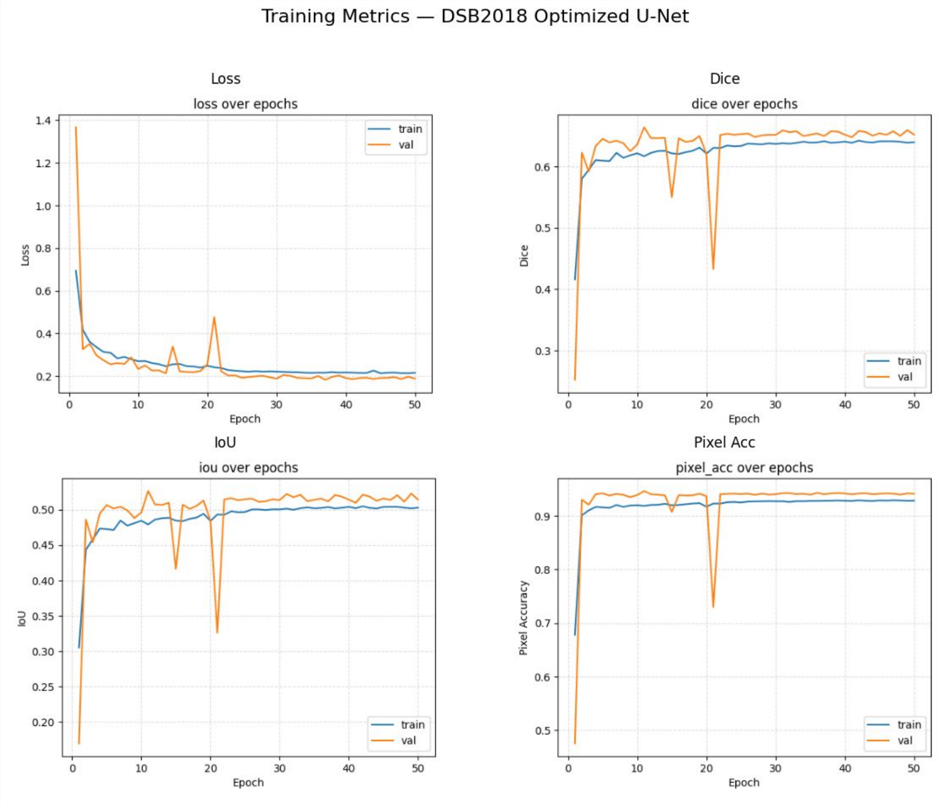


Experimental Results

03

On DSB 2018

- Dice ~66%
- Fast Convergence
- Validation IoU ~ 51%
- IoU curves also improved significantly
- Confirms DSB dataset difficulty and the need for advanced instance-aware losses for top performance



Significance of Our Results

– Scientific Significance

Shows that classical architectures still benefit greatly from modern training.

- Demonstrates that limitations in the 2015 U-Net were largely due to **training practices**, not the model design.
- Highlights how optimization choices (BatchNorm, schedulers, dropout) can have a larger impact than architectural changes
- Reinforces the value of revisiting older models with updated training pipelines.

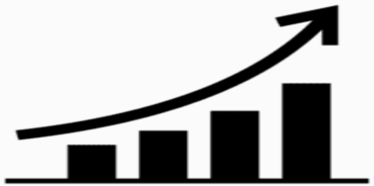
– Technical Significance

Modern optimizers meaningfully improve training stability and boundary accuracy.

Dice improved from **91.95%** → **96.2%** on PhC-U373.

IoU curves became smoother and more robust.

Demonstrates reduced noise and overfitting compared to the baseline.



– Why DSB is hard to train on U-Net?

The dataset structure breaks U-Net's original preprocessing assumptions.

- Each DSB image contains **10–15 separate instance masks**, not one.
- U-Net requires **distance-based weight maps**, which scale $O(N^2)$ with instance count and become extremely CPU-heavy.
- This makes preprocessing the bottleneck and prevents effective training on standard hardware.
- Demonstrates that classical U-Net pipelines **do not scale** to high-instance, realistic biomedical datasets without redesign.

Evaluation and Possible Experimentation

- **DSB → PHC works well** because a model trained on complex multi-instance images generalizes easily to the simpler PHC domain.
- **PHC → DSB fails** since PHC training never teaches the model to separate multiple touching nuclei or handle instance masks.
- **Both PHC models fail on DSB** because classical U-Net preprocessing (single mask, no distance maps) is incompatible with DSB's multi-instance structure.

```
=== Loading PHC validation set ===
=== Loading DSB subset ===

========================================
1) DSB2018 model → PHC validation set
========================================
Dice: 0.7737964590390524
IoU: 0.631759911775589
Pixel Accuracy: 0.9545005857944489


========================================
2) Optimized PHC model → DSB subset
========================================
Dice: 0.1984723152438665
IoU: 0.12305062013146052
Pixel Accuracy: 0.1952854374734064


========================================
3) Original U-Net → DSB subset
========================================
Dice: 0.18534868339387078
IoU: 0.1118537109073562
Pixel Accuracy: 0.1613789573001365
```

Thank You!