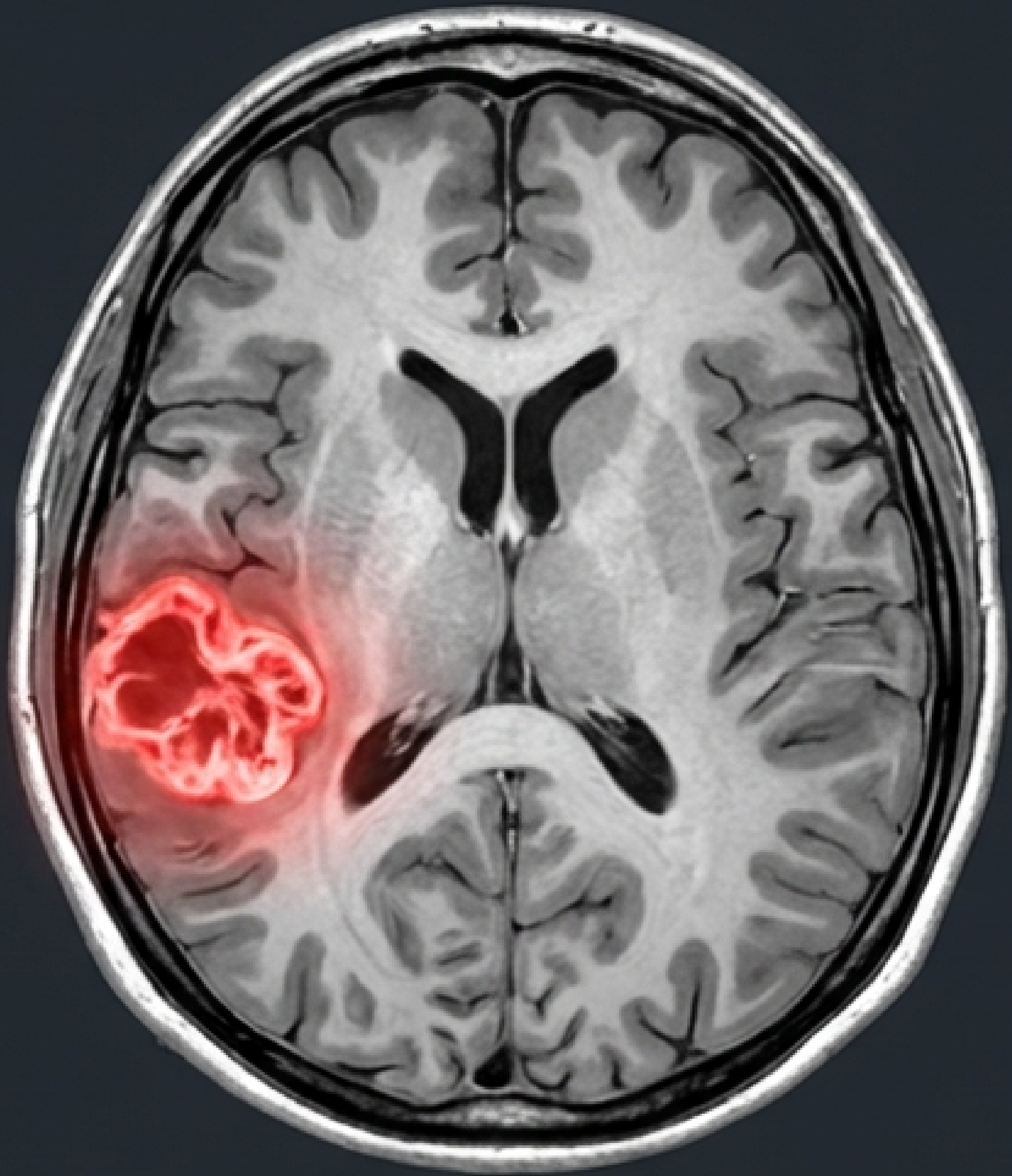


Brain MRI Tumor Segmentation

From Pixel to Prognosis: A technical deep dive into Kaggle datasets, U-Net architecture, and Dice score evaluation.

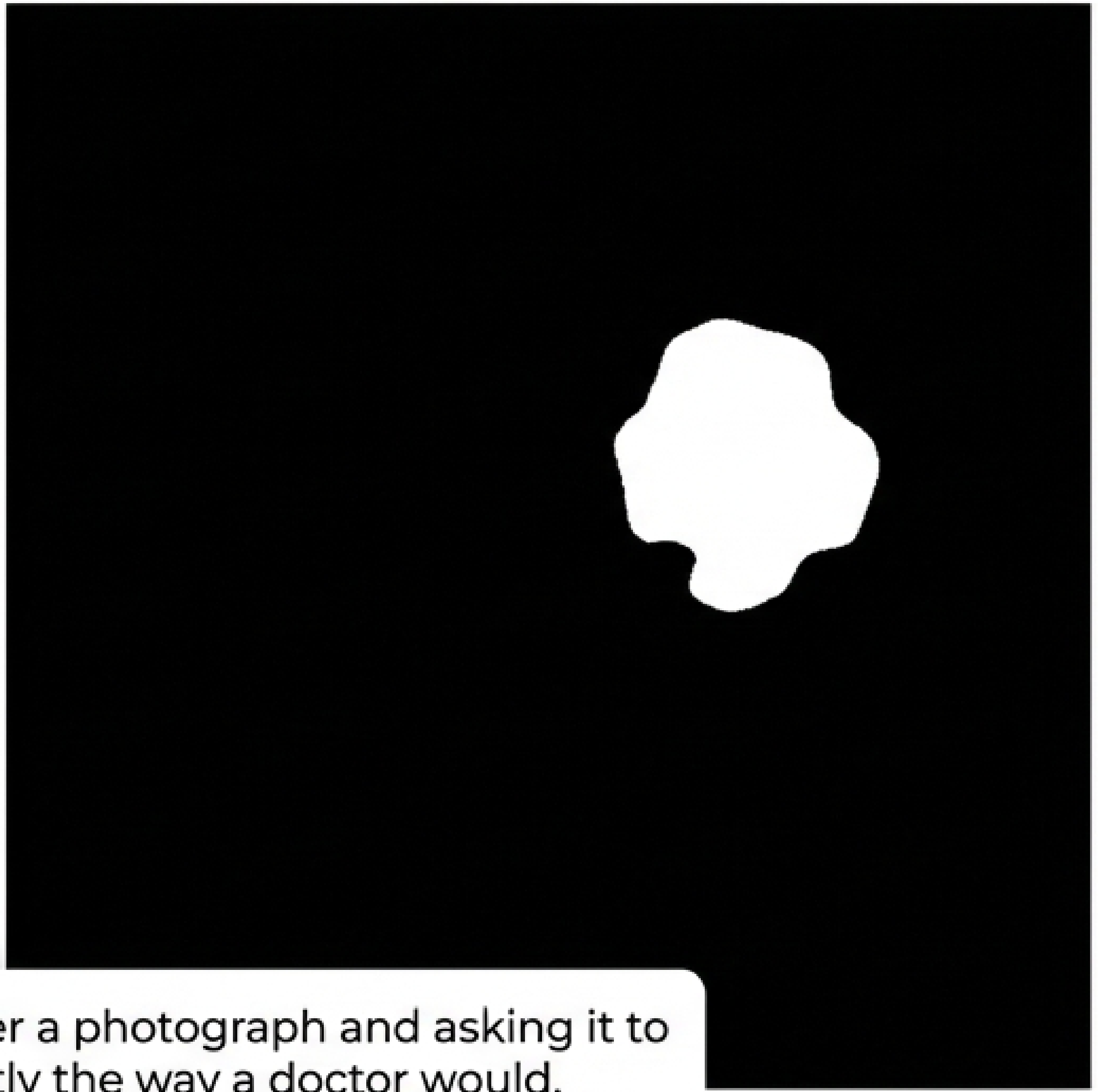
PRESENTED BY DR. KANEEKA VIDANAGE (M.PHIL, PHD)

The Objective: Coloring Inside the Lines

We are distinguishing between **Classification** (labelling an image) and **Segmentation** (partitioning an image). The goal is to label every single pixel as either **'Tumor'** (1) or **'Not Tumor'** (0).

INPUT (Grayscale)

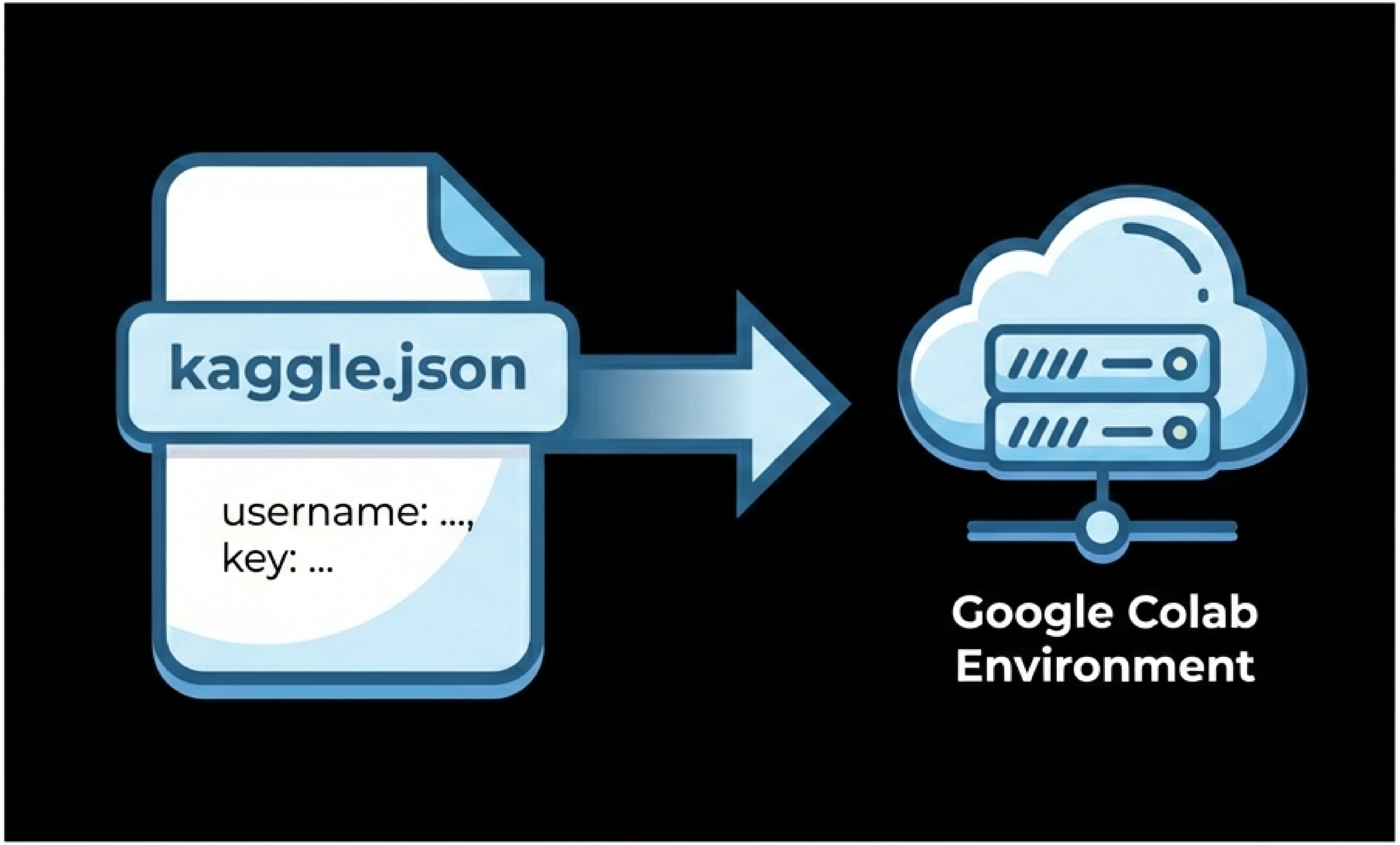
OUTPUT (Mask)



Analogy: It is like giving a computer a photograph and asking it to color inside the tumor region exactly the way a doctor would.

The Keycard: Accessing the Kaggle Library

To access the dataset programmatically, we require a credential token. This acts as a physical key to a digital door.



1. **Generate:**
   Kaggle Profile > Account > Create New API Token
2. **Download:**
   Saves 'kaggle.json' with your API Key to your local machine.
3. **Upload:**
   Transfer file to the Colab session to authenticate.

Securing the Credentials with Linux

We must place the key in the specific directory the system expects, and **lock it** so only the owner can read it.

```
!mkdir -p ~/.kaggle
>
```

Create the hidden directory (the specific drawer).

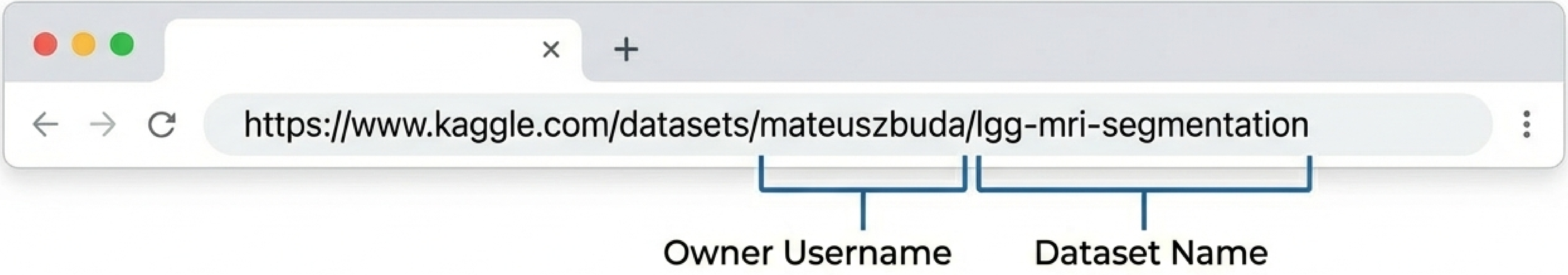
```
!cp kaggle.json ~/.kaggle/
>
```

Copy the key into the hidden directory.

```
!chmod 600 ~/.kaggle/kaggle.json
>
```

Set permissions (600). Only the owner can read/write. Kaggle refuses open keys.

Locating the Source Material

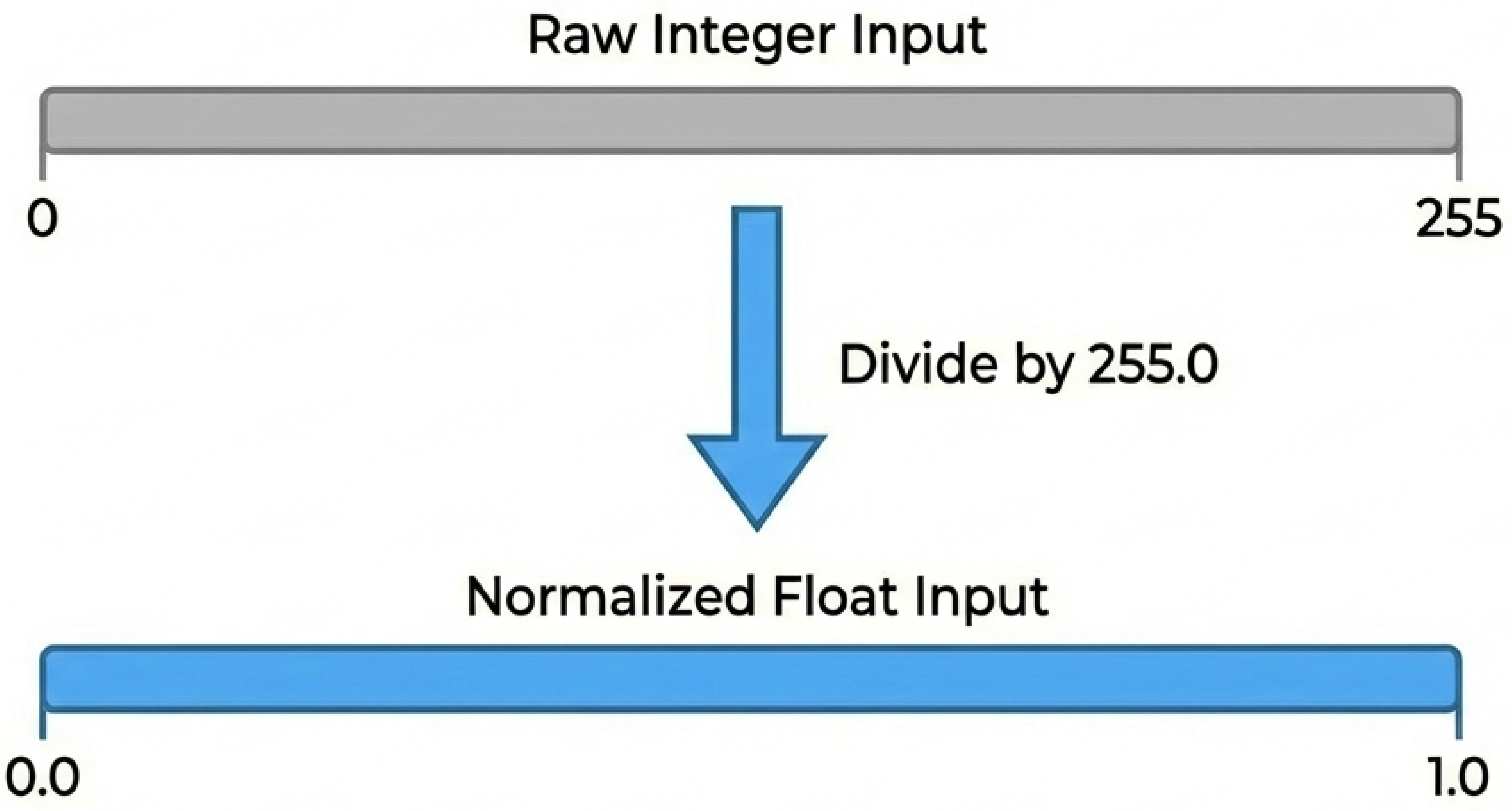



```
!kaggle datasets download -d mateuszbuda/lgg-mri-segmentation
!unzip lgg-mri-segmentation.zip
>
```

The Kaggle CLI requires the format **<owner>/<dataset>** to locate and **download** the specific **zip** file.

Standardizing the Data Language

Neural networks perform better when inputs are within a small, standard range. We **normalize** raw pixel data.



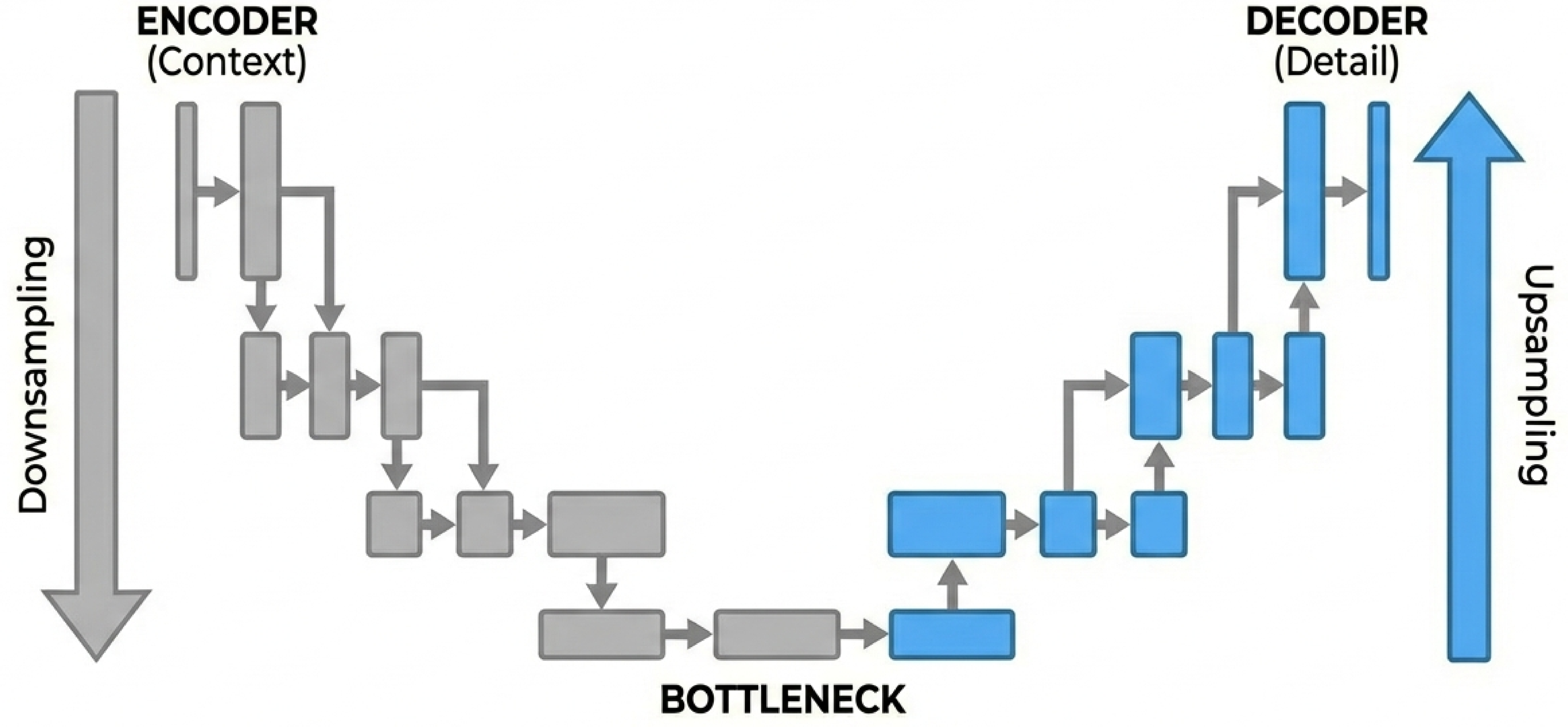
Why Normalization?

1. **Stability:** Networks learn faster with small floats.
2. **Mask Logic:** Aligns with binary logic where 0.0 is Background and 1.0 is Tumor.

Analogy: Converting a measurement system from millimeters to meters for consistency.

The Blueprint: U-Net Architecture

U-Net is a specialized convolutional network designed for biomedical image segmentation.



The Encoder: Zooming Out for Context

256x256 Input

128x128 (Pooling)

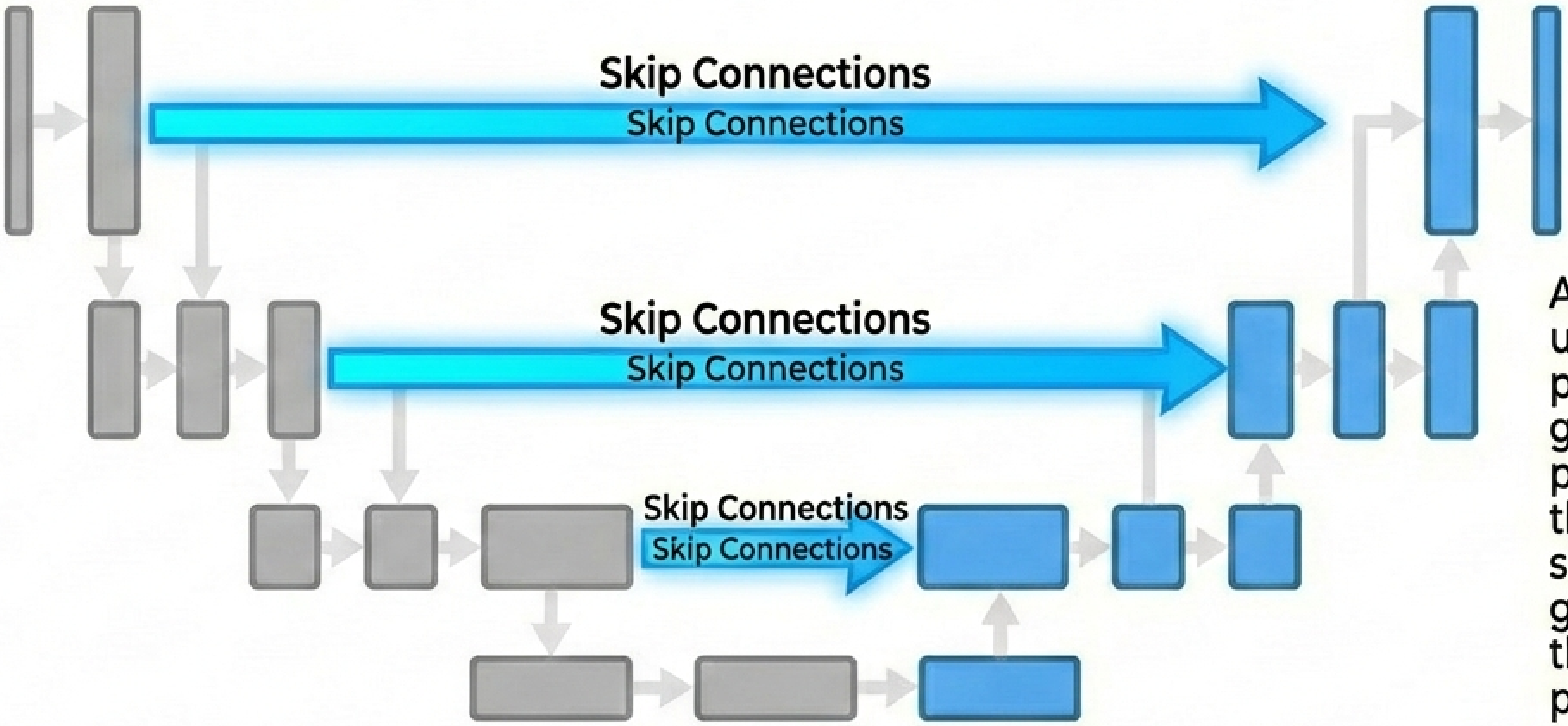
64x64 (High-Level Features)

The **Encoder** uses pooling layers to progressively reduce the image size. By sacrificing spatial resolution, the model gains ‘Big Picture’ context.

Transfer Learning Concept: This phase extracts high-level features. It answers: "Roughly where is the anomaly and what is the surrounding tissue texture?”

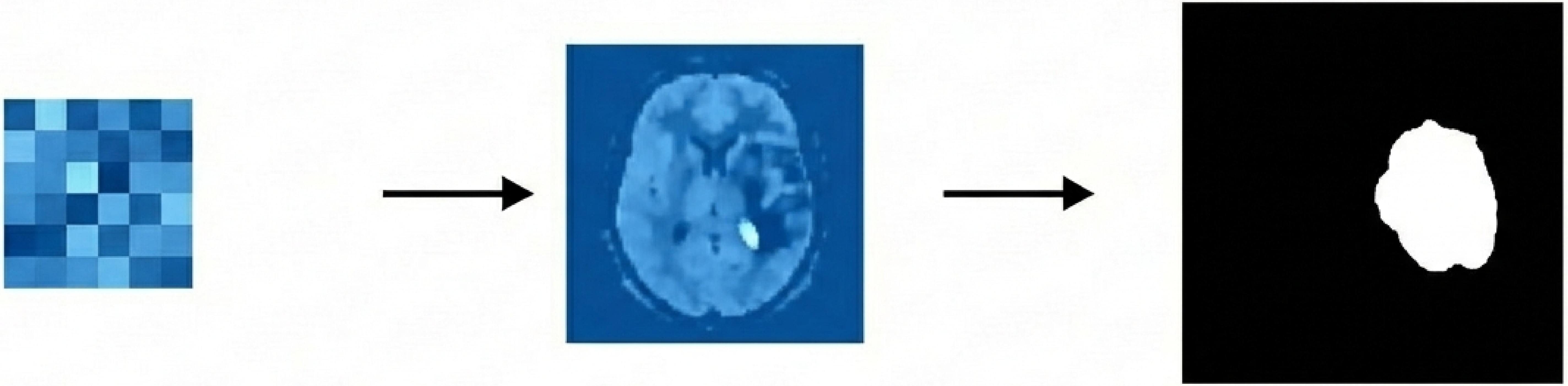
Skip Connections: The Magic Bridge

Problem: When the network zooms out, fine edge details are lost.
Solution: We physically copy information from the start to the end.



Analogy: It is like using your initial pencil sketch to guide your final painting, ensuring the edges remain sharp and do not get blurry during the coloring process.

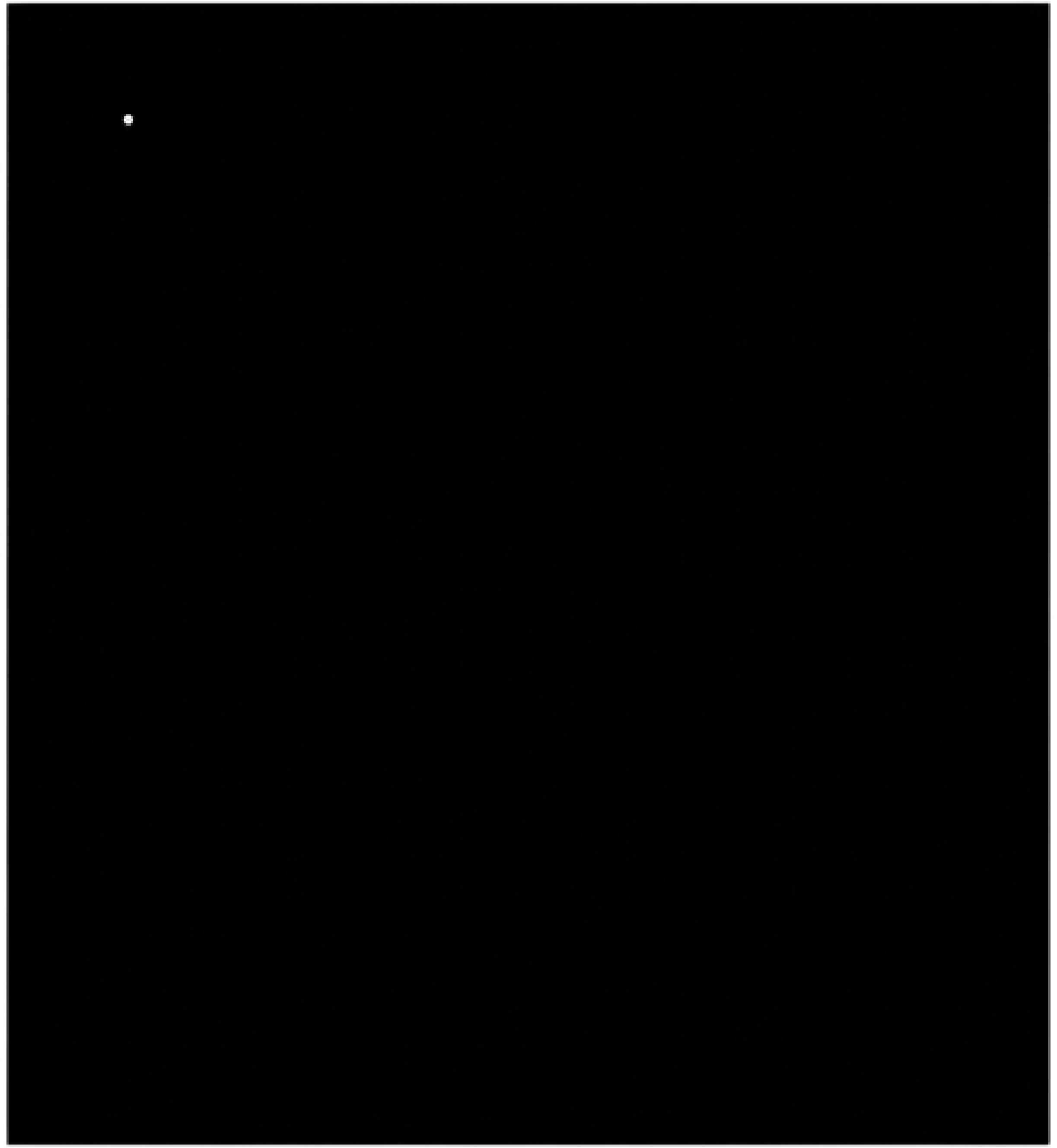
The Decoder: Zooming In for Precision



64x64 (High-Level Features) 128x128 (Upsampling) 256x256 (Segmentation Mask)

The Decoder upsamples the abstract features back to the original image size. It combines the ‘Context’ from the bottleneck with the ‘Detail’ from the Skip Connections to draw the exact boundaries of the tumor pixel-by-pixel.

The Trap of '98% Accuracy'



Scenario:
- Total Pixels: 10,000
- Tumor Pixels: 200
- Background Pixels: 9,800

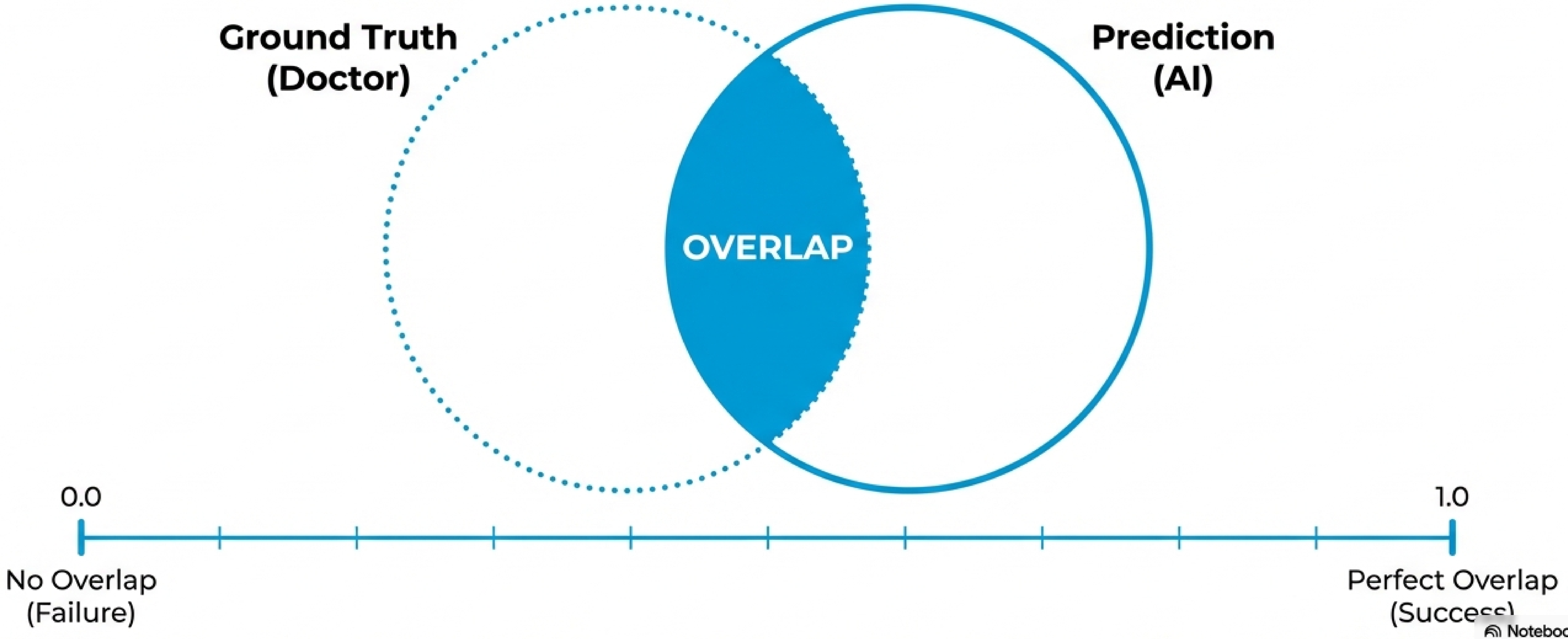
The Cheat: If the model predicts "No Tumor" (All Black), it is correct 9,800 times out of 10,000.

Accuracy = 98%.

The Verdict: Despite high accuracy, the model missed the tumor entirely. Accuracy is a useless metric for medical segmentation.

The Solution: Dice Score

We need a metric that penalizes the model for missing the tumor, **regardless** of how much background there is.



Calculating the Dice Score

Ground Truth (T)

	1	1		
	1	1		

T = 4

Prediction (P)

		1	1	
		1	1	

P = 4

Overlap (O)

		1	1	
		1		

O = 3

Dice = (2 × Overlap) / (Total Predicted + Total True)

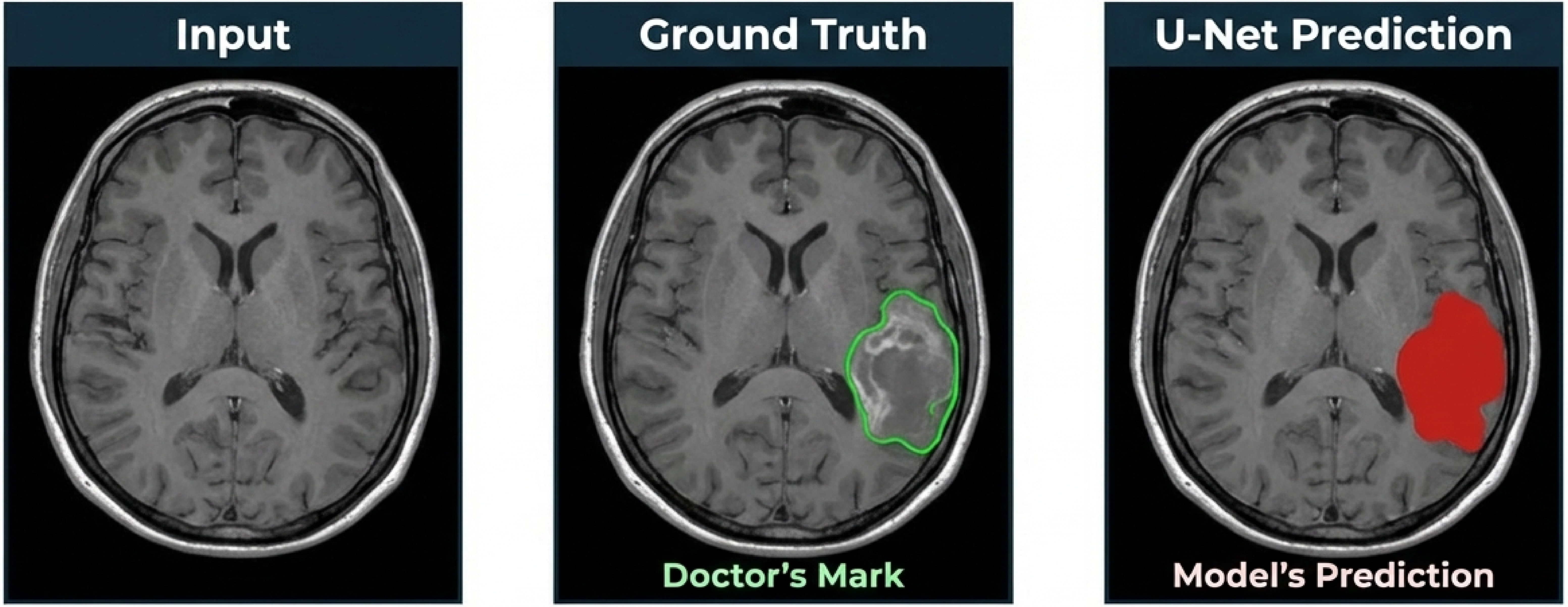
Dice = (2 × 3) / (4 + 4)

Dice = 6 / 8 = 0.75

Dice Loss = 1 - 0.75 = **0.25** (Lower is better)

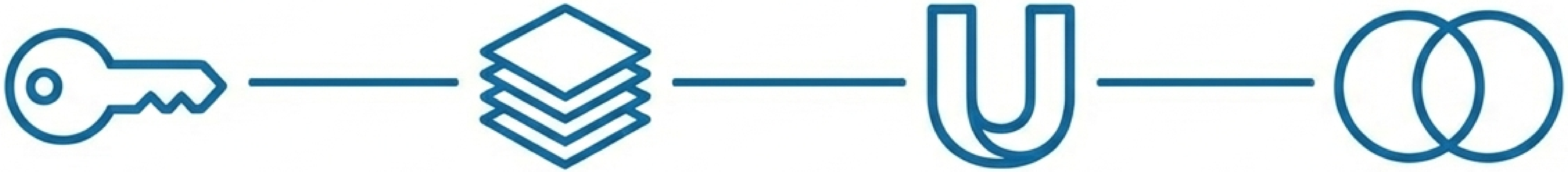
Visual Validation: Ground Truth vs. Prediction

The final test is a visual inspection.



In a successful model, the **Red** region in Panel 3 should perfectly match the **Green** outline in Panel 2.

Summary: The Journey from Data to Diagnosis



Access	Prep	Model	Eval
kaggle.json unlocks the data.	Normalization & Resizing.	U-Net Context & Detail.	Dice Score checks overlap.

By combining secure data pipelines, specialized architecture, and rigorous overlap metrics, we turn raw pixels into life-saving insights.