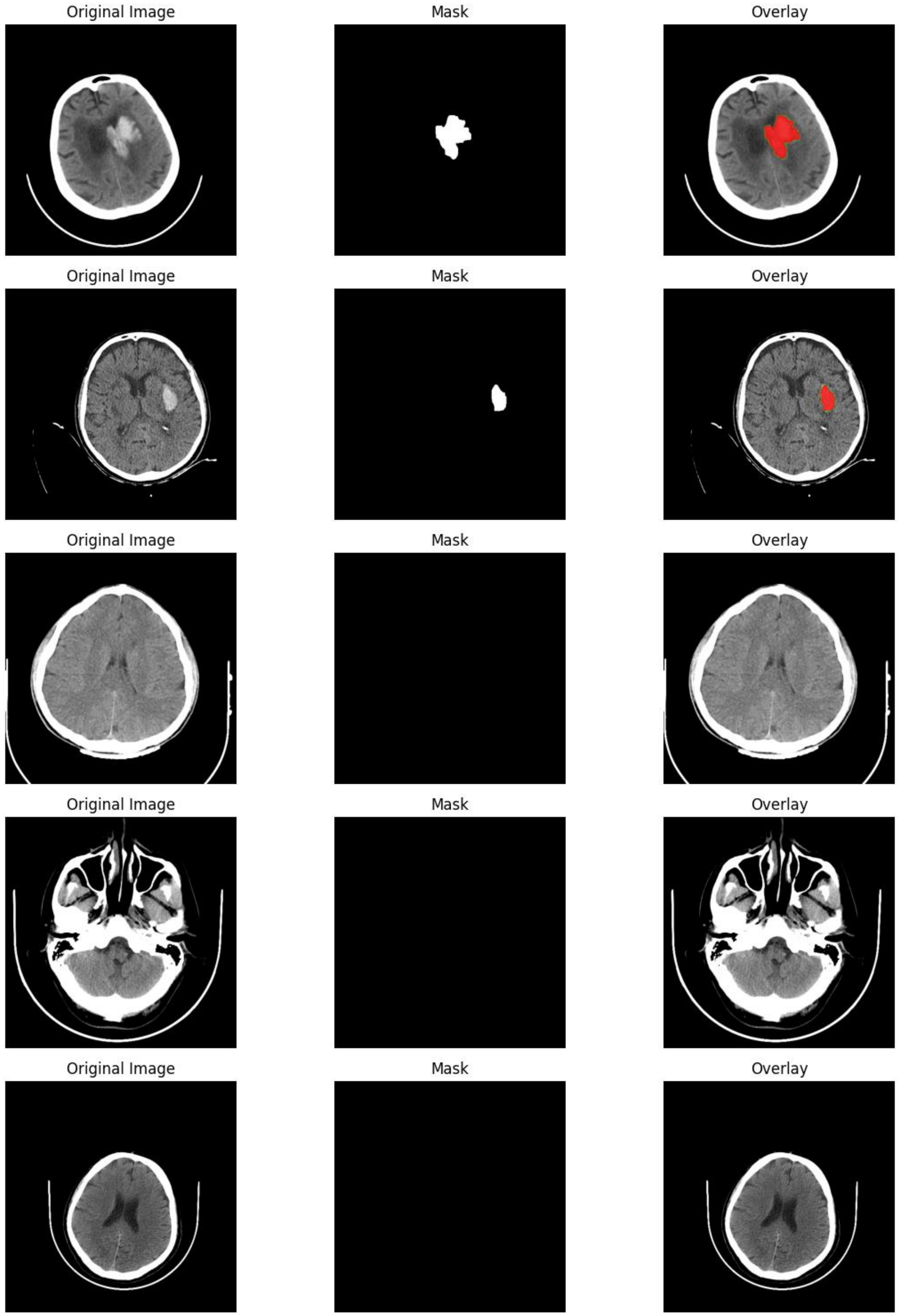
Early Detection and Prediction of Brain Stroke Severity using Unet with Multihead Attention Mechanism

```
import numpy as np
import pandas as pd
import os

import matplotlib.pyplot as plt
from PIL import Image
from sklearn.model_selection import train_test_split
import tensorflow as tf
from tensorflow.keras import layers, models

base_path = "/kaggle/input/inme-veri-seti-stroke-dataset/İNME VERİ
SETİ/YarısmaVeriSeti_1_Oturum"
png_path = os.path.join(base_path, "PNG")
masks_path = os.path.join(base_path, "MASKS")
overlay_path = os.path.join(base_path, "OVERLAY")

import cv2

png_files = sorted(os.listdir(png_path))[:5]
mask_files = sorted(os.listdir(masks_path))[:5]
overlay_files = sorted(os.listdir(overlay_path))[:5]

fig, axes = plt.subplots(5, 3, figsize=(12, 15))

for i in range(5):

    img = cv2.imread(os.path.join(png_path, png_files[i]))
    mask = cv2.imread(os.path.join(masks_path, mask_files[i]),
```

```
cv2.IMREAD_GRAYSCALE)
    overlay = cv2.imread(os.path.join(overlay_path, overlay_files[i]))

    img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
    overlay = cv2.cvtColor(overlay, cv2.COLOR_BGR2RGB)

    axes[i, 0].imshow(img)
    axes[i, 0].set_title("Original Image")
    axes[i, 0].axis("off")

    axes[i, 1].imshow(mask, cmap="gray")
    axes[i, 1].set_title("Mask")
    axes[i, 1].axis("off")

    axes[i, 2].imshow(overlay)
    axes[i, 2].set_title("Overlay")
    axes[i, 2].axis("off")

plt.tight_layout()
plt.show()
```


Original Image
Mask
Overlay
Original Image
Mask
Overlay
Original Image
Mask
Overlay
Original Image
Mask
Overlay
Original Image
Mask
Overlay

```
def load_images_and_masks(image_folder, mask_folder, image_size=(256, 256)):
    images = []
    masks = []
    image_files = sorted(os.listdir(image_folder))
    mask_files = sorted(os.listdir(mask_folder))

    for img_file, mask_file in zip(image_files, mask_files):
        img_path = os.path.join(image_folder, img_file)
        mask_path = os.path.join(mask_folder, mask_file)

        img = Image.open(img_path).convert("L").resize(image_size)
        mask = Image.open(mask_path).convert("L").resize(image_size)

        img = np.array(img) / 255.0
        mask = np.array(mask) / 255.0

        images.append(np.expand_dims(img, axis=-1))
        masks.append(np.expand_dims(mask, axis=-1))

    return np.array(images), np.array(masks)

images, masks = load_images_and_masks(png_path, masks_path)

import tensorflow as tf

gpus = tf.config.list_physical_devices('GPU')
if gpus:
    try:
        for gpu in gpus:
            tf.config.experimental.set_memory_growth(gpu, True)
        print("GPUs Available:", gpus)
    except RuntimeError as e:
        print(e)
else:
    print("No GPU found. Running on CPU.")

GPUs Available: [PhysicalDevice(name='/physical_device:GPU:0',
device_type='GPU'), PhysicalDevice(name='/physical_device:GPU:1',
device_type='GPU')]

X_train, X_val, y_train, y_val = train_test_split(images, masks,
test_size=0.2, random_state=42)

def unet_model(input_size=(256, 256, 1)):
    inputs = tf.keras.Input(input_size)


    conv1 = layers.Conv2D(64, 3, activation="relu", padding="same")(inputs)
    conv1 = layers.Conv2D(64, 3, activation="relu", padding="same")(conv1)
    pool1 = layers.MaxPooling2D(pool_size=(2, 2))(conv1)
```

```python
    conv2 = layers.Conv2D(128, 3, activation="relu", padding="same")(pool1)
    conv2 = layers.Conv2D(128, 3, activation="relu", padding="same")(conv2)
    pool2 = layers.MaxPooling2D(pool_size=(2, 2))(conv2)

    conv3 = layers.Conv2D(256, 3, activation="relu", padding="same")(pool2)
    conv3 = layers.Conv2D(256, 3, activation="relu", padding="same")(conv3)

    up1 = layers.UpSampling2D(size=(2, 2))(conv3)
    concat1 = layers.concatenate([conv2, up1], axis=-1)
    conv4 = layers.Conv2D(128, 3, activation="relu", padding="same")(concat1)
    conv4 = layers.Conv2D(128, 3, activation="relu", padding="same")(conv4)

    up2 = layers.UpSampling2D(size=(2, 2))(conv4)
    concat2 = layers.concatenate([conv1, up2], axis=-1)
    conv5 = layers.Conv2D(64, 3, activation="relu", padding="same")(concat2)
    conv5 = layers.Conv2D(64, 3, activation="relu", padding="same")(conv5)

    outputs = layers.Conv2D(1, 1, activation="sigmoid")(conv5)

    model = models.Model(inputs, outputs)
    return model

model = unet_model(input_size=(256, 256, 1))
model.compile(optimizer="adam", loss="binary_crossentropy",
metrics=["accuracy"])
model.summary()
```

```
Model: "functional_2"

┏━━━━━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━━━┳━━━━━━━
━━━━━━━━━━━━━━━━━━━┓
┃ Layer (type)               ┃ Output Shape            ┃         Param # ┃
Connected to          ┃
┡━━━━━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━━━╇━━━━━━━
━━━━━━━━━━━━━━━━━━━┩
│ input_layer_2              │ (None, 256, 256, 1)     │               0 │ -

│ (InputLayer)               │                         │                 │

├────────────────────────────┼─────────────────────────┼─────────────────┼───────
───────────────────┤
│ conv2d_22 (Conv2D)         │ (None, 256, 256, 64)    │             640 │
input_layer_2[0][0]   │
├────────────────────────────┼─────────────────────────┼─────────────────┼───────
───────────────────┤
│ conv2d_23 (Conv2D)         │ (None, 256, 256, 64)    │          36,928 │
conv2d_22[0][0]       │
├────────────────────────────┼─────────────────────────┼─────────────────┼───────
───────────────────┤
```

max_pooling2d_4 (MaxPooling2D)	(None, 128, 128, 64)	0	conv2d_23[0][0]
conv2d_24 (Conv2D)	(None, 128, 128, 128)	73,856	max_pooling2d_4[0][0]
conv2d_25 (Conv2D)	(None, 128, 128, 128)	147,584	conv2d_24[0][0]
max_pooling2d_5 (MaxPooling2D)	(None, 64, 64, 128)	0	conv2d_25[0][0]
conv2d_26 (Conv2D)	(None, 64, 64, 256)	295,168	max_pooling2d_5[0][0]
conv2d_27 (Conv2D)	(None, 64, 64, 256)	590,080	conv2d_26[0][0]
up_sampling2d_4 (UpSampling2D)	(None, 128, 128, 256)	0	conv2d_27[0][0]
concatenate_4 (Concatenate)	(None, 128, 128, 384)	0	conv2d_25[0][0], up_sampling2d_4[0][0]
conv2d_28 (Conv2D)	(None, 128, 128, 128)	442,496	concatenate_4[0][0]
conv2d_29 (Conv2D)	(None, 128, 128, 128)	147,584	conv2d_28[0][0]
up_sampling2d_5	(None, 256, 256, 128)	0	conv2d_29[0][0]

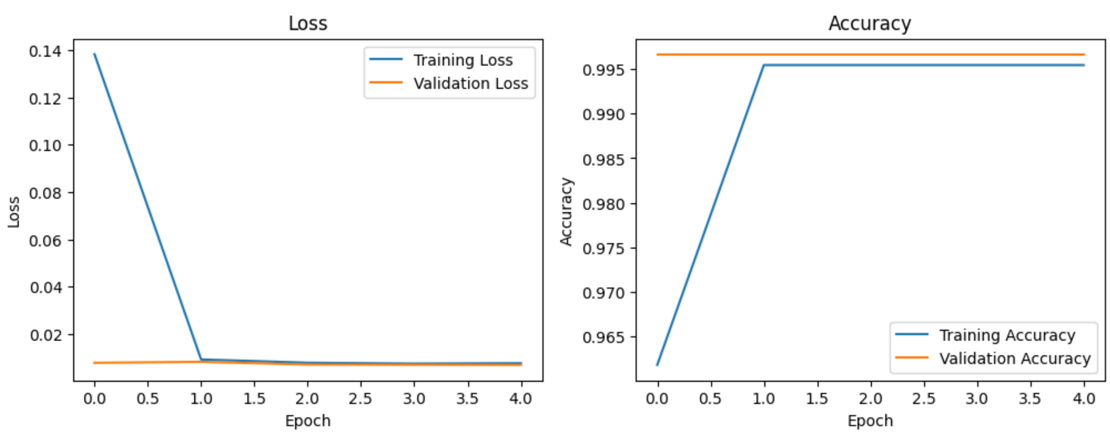
```
│ (UpSampling2D)              │                         │                │
│
├─────────────────────────────┼─────────────────────────┼────────────────┼───────
──────────────────┤
│ concatenate_5               │ (None, 256, 256, 192)   │              0 │
conv2d_23[0][0],        │
│ (Concatenate)               │                         │                │
up_sampling2d_5[0][0]   │
├─────────────────────────────┼─────────────────────────┼────────────────┼───────
──────────────────┤
│ conv2d_30 (Conv2D)          │ (None, 256, 256, 64)    │        110,656 │
concatenate_5[0][0]     │
├─────────────────────────────┼─────────────────────────┼────────────────┼───────
──────────────────┤
│ conv2d_31 (Conv2D)          │ (None, 256, 256, 64)    │         36,928 │
conv2d_30[0][0]         │
├─────────────────────────────┼─────────────────────────┼────────────────┼───────
──────────────────┤
│ conv2d_32 (Conv2D)          │ (None, 256, 256, 1)     │             65 │
conv2d_31[0][0]         │
└─────────────────────────────┴─────────────────────────┴────────────────┴───────
──────────────────┘

 Total params: 1,881,985 (7.18 MB)

 Trainable params: 1,881,985 (7.18 MB)

 Non-trainable params: 0 (0.00 B)
```

```python
history = model.fit(
    X_train, y_train,
    validation_data=(X_val, y_val),
    batch_size=8,
    epochs=5,
    verbose=1
)
```

```
Epoch 1/5
20/20 ━━━━━━━━━━━━━━━━━━━━11s 328ms/step - accuracy: 0.8792 - loss: 0.3010 -
val_accuracy: 0.9967 - val_loss: 0.0078
Epoch 2/5
20/20 ━━━━━━━━━━━━━━━━━━━━6s 297ms/step - accuracy: 0.9959 - loss: 0.0098 -
val_accuracy: 0.9967 - val_loss: 0.0082
Epoch 3/5
20/20 ━━━━━━━━━━━━━━━━━━━━6s 302ms/step - accuracy: 0.9955 - loss: 0.0076 -
val_accuracy: 0.9967 - val_loss: 0.0070
Epoch 4/5
20/20 ━━━━━━━━━━━━━━━━━━━━6s 307ms/step - accuracy: 0.9959 - loss: 0.0060 -
val_accuracy: 0.9967 - val_loss: 0.0069
Epoch 5/5
```

```
20/20 ━━━━━━━━━━━━━━━━━━━━6s 309ms/step - accuracy: 0.9958 - loss: 0.0070 -
val_accuracy: 0.9967 - val_loss: 0.0069
```

```python
def plot_training_history(history):
    plt.figure(figsize=(12, 4))

    plt.subplot(1, 2, 1)
    plt.plot(history.history["loss"], label="Training Loss")
    plt.plot(history.history["val_loss"], label="Validation Loss")
    plt.title("Loss")
    plt.xlabel("Epoch")
    plt.ylabel("Loss")
    plt.legend()

    plt.subplot(1, 2, 2)
    plt.plot(history.history["accuracy"], label="Training Accuracy")
    plt.plot(history.history["val_accuracy"], label="Validation Accuracy")
    plt.title("Accuracy")
    plt.xlabel("Epoch")
    plt.ylabel("Accuracy")
    plt.legend()

    plt.show()

plot_training_history(history)
```




```python
def multihead_attention_block(inputs, key_dim, num_heads):
    """Adds a multi-head attention block."""
    attention = layers.MultiHeadAttention(num_heads=num_heads,
key_dim=key_dim)(inputs, inputs)
    attention = layers.LayerNormalization(epsilon=1e-6)(attention + inputs)
    return attention

def unet_model(input_size=(256, 256, 1)):
    inputs = tf.keras.Input(input_size)
```

```
    conv1 = layers.Conv2D(64, 3, activation="relu", padding="same")(inputs)
    conv1 = layers.Conv2D(64, 3, activation="relu", padding="same")(conv1)
    pool1 = layers.MaxPooling2D(pool_size=(2, 2))(conv1)

    conv2 = layers.Conv2D(128, 3, activation="relu", padding="same")(pool1)
    conv2 = layers.Conv2D(128, 3, activation="relu", padding="same")(conv2)
    pool2 = layers.MaxPooling2D(pool_size=(2, 2))(conv2)

    conv3 = layers.Conv2D(256, 3, activation="relu", padding="same")(pool2)
    conv3 = layers.Conv2D(256, 3, activation="relu", padding="same")(conv3)

    reshaped_conv3 = layers.Reshape((-1, 256))(conv3)
    attention_conv3 = multihead_attention_block(reshaped_conv3,
key_dim=256//4, num_heads=4)
    attention_conv3 = layers.Reshape((conv3.shape[1], conv3.shape[2],
256))(attention_conv3)

    up1 = layers.UpSampling2D(size=(2, 2))(attention_conv3)
    concat1 = layers.concatenate([conv2, up1], axis=-1)
    conv4 = layers.Conv2D(128, 3, activation="relu", padding="same")(concat1)
    conv4 = layers.Conv2D(128, 3, activation="relu", padding="same")(conv4)

    up2 = layers.UpSampling2D(size=(2, 2))(conv4)
    concat2 = layers.concatenate([conv1, up2], axis=-1)
    conv5 = layers.Conv2D(64, 3, activation="relu", padding="same")(concat2)
    conv5 = layers.Conv2D(64, 3, activation="relu", padding="same")(conv5)

    outputs = layers.Conv2D(1, 1, activation="sigmoid")(conv5)

    model = models.Model(inputs, outputs)
    return model

model = unet_model(input_size=(256, 256, 1))
model.compile(optimizer="adam", loss="binary_crossentropy",
metrics=["accuracy"])
model.summary()
```

```
Model: "functional_3"
```

Layer (type)	Output Shape	Param #	Connected to
input_layer_3 (InputLayer)	(None, 256, 256, 1)	0	-

```
──────────────────┤
│ conv2d_33 (Conv2D)          │ (None, 256, 256, 64)   │            640 │
input_layer_3[0][0]    │
├─────────────────────────────┼────────────────────────┼────────────────┼──────
──────────────────┤
│ conv2d_34 (Conv2D)          │ (None, 256, 256, 64)   │         36,928 │
conv2d_33[0][0]        │
├─────────────────────────────┼────────────────────────┼────────────────┼──────
──────────────────┤
│ max_pooling2d_6             │ (None, 128, 128, 64)   │              0 │
conv2d_34[0][0]        │
│ (MaxPooling2D)              │                        │                │
│
├─────────────────────────────┼────────────────────────┼────────────────┼──────
──────────────────┤
│ conv2d_35 (Conv2D)          │ (None, 128, 128, 128)  │         73,856 │
max_pooling2d_6[0][0]  │
├─────────────────────────────┼────────────────────────┼────────────────┼──────
──────────────────┤
│ conv2d_36 (Conv2D)          │ (None, 128, 128, 128)  │        147,584 │
conv2d_35[0][0]        │
├─────────────────────────────┼────────────────────────┼────────────────┼──────
──────────────────┤
│ max_pooling2d_7             │ (None, 64, 64, 128)    │              0 │
conv2d_36[0][0]        │
│ (MaxPooling2D)              │                        │                │
│
├─────────────────────────────┼────────────────────────┼────────────────┼──────
──────────────────┤
│ conv2d_37 (Conv2D)          │ (None, 64, 64, 256)    │        295,168 │
max_pooling2d_7[0][0]  │
├─────────────────────────────┼────────────────────────┼────────────────┼──────
──────────────────┤
│ conv2d_38 (Conv2D)          │ (None, 64, 64, 256)    │        590,080 │
conv2d_37[0][0]        │
├─────────────────────────────┼────────────────────────┼────────────────┼──────
──────────────────┤
│ reshape_2 (Reshape)         │ (None, 4096, 256)      │              0 │
conv2d_38[0][0]        │
├─────────────────────────────┼────────────────────────┼────────────────┼──────
──────────────────┤
│ multi_head_attention_1      │ (None, 4096, 256)      │        263,168 │
reshape_2[0][0],       │
│ (MultiHeadAttention)        │                        │                │
reshape_2[0][0]        │
├─────────────────────────────┼────────────────────────┼────────────────┼──────
──────────────────┤
│ add_1 (Add)                 │ (None, 4096, 256)      │              0 │
multi_head_attention_… │
│                             │                        │                │
```

```
reshape_2[0][0]          │
├───────────────────────────┼─────────────────────────┼────────────────┼──────
──────────────────┤
│ layer_normalization_1     │ (None, 4096, 256)       │            512 │
add_1[0][0]              │
│ (LayerNormalization)      │                         │                │
│
├───────────────────────────┼─────────────────────────┼────────────────┼──────
──────────────────┤
│ reshape_3 (Reshape)       │ (None, 64, 64, 256)     │              0 │
layer_normalization_1… │
├───────────────────────────┼─────────────────────────┼────────────────┼──────
──────────────────┤
│ up_sampling2d_6           │ (None, 128, 128, 256)   │              0 │
reshape_3[0][0]          │
│ (UpSampling2D)            │                         │                │
│
├───────────────────────────┼─────────────────────────┼────────────────┼──────
──────────────────┤
│ concatenate_6             │ (None, 128, 128, 384)   │              0 │
conv2d_36[0][0],         │
│ (Concatenate)             │                         │                │
up_sampling2d_6[0][0]    │
├───────────────────────────┼─────────────────────────┼────────────────┼──────
──────────────────┤
│ conv2d_39 (Conv2D)        │ (None, 128, 128, 128)   │        442,496 │
concatenate_6[0][0]      │
├───────────────────────────┼─────────────────────────┼────────────────┼──────
──────────────────┤
│ conv2d_40 (Conv2D)        │ (None, 128, 128, 128)   │        147,584 │
conv2d_39[0][0]          │
├───────────────────────────┼─────────────────────────┼────────────────┼──────
──────────────────┤
│ up_sampling2d_7           │ (None, 256, 256, 128)   │              0 │
conv2d_40[0][0]          │
│ (UpSampling2D)            │                         │                │
│
├───────────────────────────┼─────────────────────────┼────────────────┼──────
──────────────────┤
│ concatenate_7             │ (None, 256, 256, 192)   │              0 │
conv2d_34[0][0],         │
│ (Concatenate)             │                         │                │
up_sampling2d_7[0][0]    │
├───────────────────────────┼─────────────────────────┼────────────────┼──────
──────────────────┤
│ conv2d_41 (Conv2D)        │ (None, 256, 256, 64)    │        110,656 │
concatenate_7[0][0]      │
├───────────────────────────┼─────────────────────────┼────────────────┼──────
──────────────────┤
│ conv2d_42 (Conv2D)        │ (None, 256, 256, 64)    │         36,928 │
```

```
conv2d_41[0][0]        │
├───────────────────────┼─────────────────────────┼────────────────┼───────
────────────────────┤
│ conv2d_43 (Conv2D)     │ (None, 256, 256, 1)     │             65 │
conv2d_42[0][0]        │
└───────────────────────┴─────────────────────────┴────────────────┴───────
────────────────────┘

 Total params: 2,145,665 (8.19 MB)

 Trainable params: 2,145,665 (8.19 MB)

 Non-trainable params: 0 (0.00 B)
```

```
history = model.fit(
    X_train, y_train,
    validation_data=(X_val, y_val),
    batch_size=8,
    epochs=5,
    verbose=1
)
```

```
Epoch 1/5
20/20 ━━━━━━━━━━━━━━━━━━━━━19s 648ms/step - accuracy: 0.9924 - loss: 0.1135 -
val_accuracy: 0.9967 - val_loss: 0.0085
Epoch 2/5
20/20 ━━━━━━━━━━━━━━━━━━━━━12s 607ms/step - accuracy: 0.9960 - loss: 0.0095 -
val_accuracy: 0.9967 - val_loss: 0.0082
Epoch 3/5
20/20 ━━━━━━━━━━━━━━━━━━━━━12s 601ms/step - accuracy: 0.9963 - loss: 0.0070 -
val_accuracy: 0.9967 - val_loss: 0.0087
Epoch 4/5
20/20 ━━━━━━━━━━━━━━━━━━━━━12s 597ms/step - accuracy: 0.9958 - loss: 0.0080 -
val_accuracy: 0.9967 - val_loss: 0.0082
Epoch 5/5
20/20 ━━━━━━━━━━━━━━━━━━━━━12s 595ms/step - accuracy: 0.9969 - loss: 0.0066 -
val_accuracy: 0.9967 - val_loss: 0.0075
```

```
def plot_training_history(history):
    plt.figure(figsize=(12, 4))

    plt.subplot(1, 2, 1)
    plt.plot(history.history["loss"], label="Training Loss")
    plt.plot(history.history["val_loss"], label="Validation Loss")
    plt.title("Loss")
    plt.xlabel("Epoch")
    plt.ylabel("Loss")
    plt.legend()

    plt.subplot(1, 2, 2)
    plt.plot(history.history["accuracy"], label="Training Accuracy")
```

```python
    plt.plot(history.history["val_accuracy"], label="Validation Accuracy")
    plt.title("Accuracy")
    plt.xlabel("Epoch")
    plt.ylabel("Accuracy")
    plt.legend()

    plt.show()

plot_training_history(history)
```