Documentation: U-Net for Semantic Segmentation with Custom Loss Functions

1. Data Import and Preparation

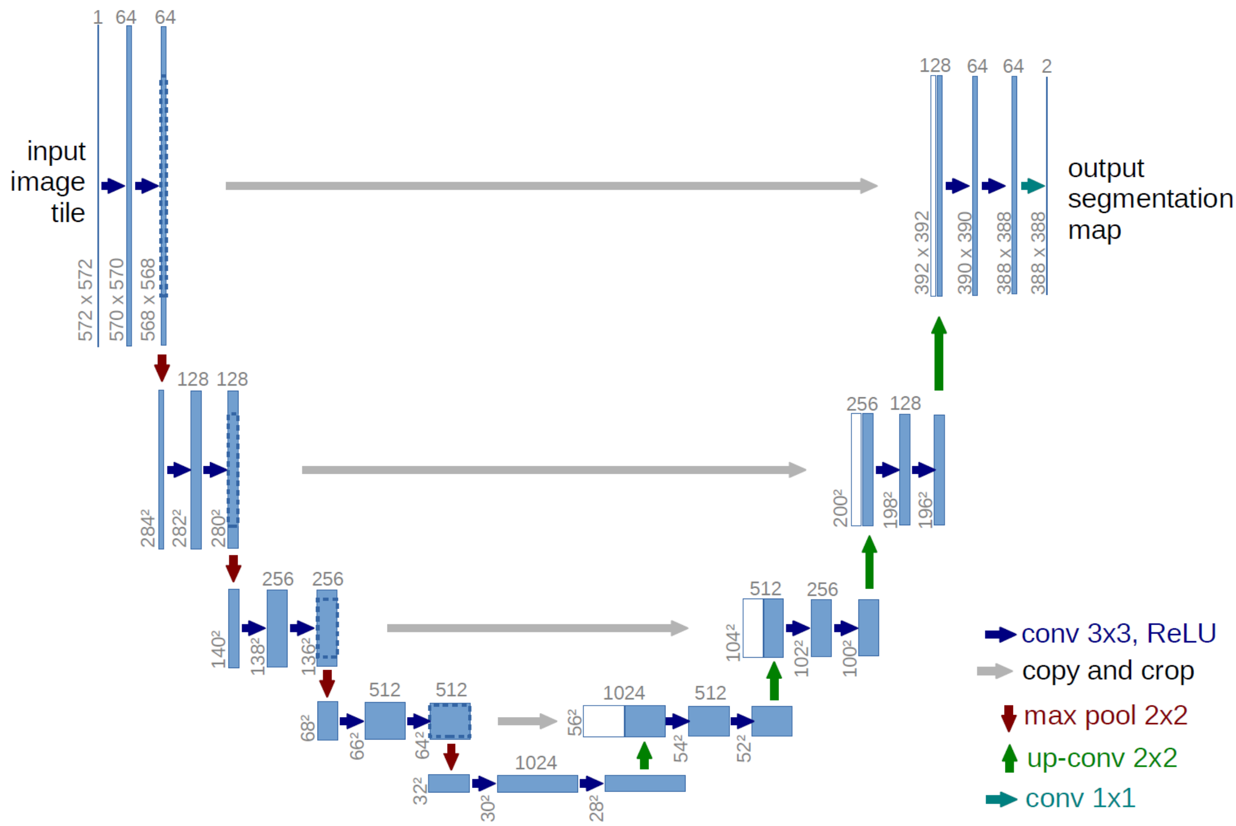
Dataset Downloading: The dataset is downloaded from Kaggle using the provided mapping of dataset names to download URLs. The dataset is extracted and organized into appropriate directories for images and masks.

Data Generators: Image and mask data generators are created using `ImageDataGenerator` from Keras. These generators facilitate loading batches of images and masks during training, which is crucial for efficient training of large datasets.

2. U-Net Model Definition

U-Net Architectures: Two implementations of the U-Net architecture, `unet_model` and `build_unet`, are provided. Both models consist of:

- **Encoder Path:** A series of convolutional and max-pooling layers to downsample the input image and capture contextual information.
- **Decoder Path:** Convolutional and upsampling layers to restore the spatial resolution of the image and produce the segmentation mask.
- **Skip Connections:** Concatenate feature maps from the encoder to the decoder to combine low-level and high-level features, facilitating accurate localization.



Model Output: The final layer uses a sigmoid activation function to produce binary segmentation masks.

3. Custom Loss Functions

Focal Loss: Focal loss is designed to address class imbalance by focusing on hard-to-classify examples.

Partial Cross Entropy Loss: Partial cross-entropy loss penalizes the misclassification of the foreground classes more heavily than the background class. This is crucial in semantic segmentation tasks where the background often dominates the image.

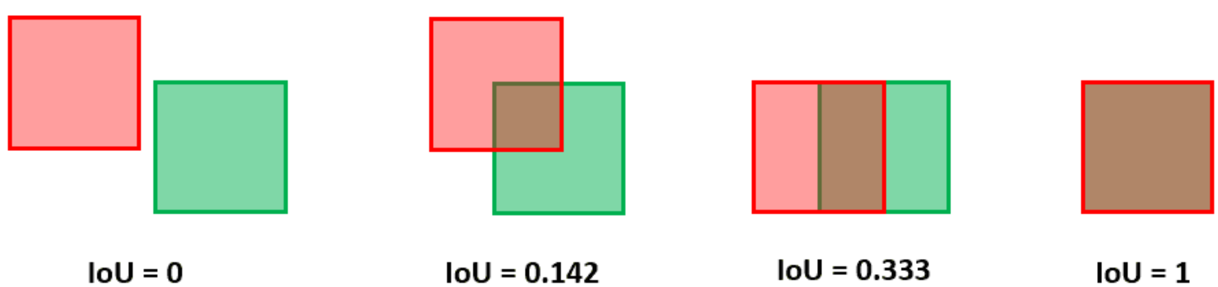
A special loss function (called partial CE loss) is designed so that point labeling can be used to train the segmentation model

$$pfCE = \frac{\sum(Focal\,loss(pre, GT) \times MASK_{labeled})}{\sum MASK_{labeled}}$$

Combined Loss Function: A combined loss function, `combined_loss`, is defined as the sum of the focal loss and the partial cross-entropy loss. This loss function is used for model compilation, aiming to balance the learning process between foreground and background classes.

4. Model Training

Compilation: The model is compiled using the Adam optimizer and the combined loss function. Metrics such as Intersection over Union (IoU) and accuracy are used to evaluate the model's performance.



Training: The model is trained using the `fit` method with early stopping and model checkpoint callbacks to prevent overfitting and save the best model based on validation loss.

1.5 Model Training

The model is compiled using the Adam optimizer and the combined loss function. Training is performed using the `fit` method, with early stopping and model checkpoint callbacks.

1.6 Evaluation and Inference

Learning curves are plotted to visualize model performance during training. Random samples from the validation set are used for inference, and original images, ground truth masks, and predicted masks are visualized side by side for comparison.

2. Experiment Design to Explore Factors Affecting Performance

Experiment 1: Effect of Learning Rate on Model Performance

2.1 Purpose/Hypothesis

Purpose: To investigate how different learning rates affect the performance of the U-Net model for semantic segmentation.

Hypothesis: The learning rate will significantly impact the model's ability to learn and generalize. A too high learning rate might cause the model to converge too quickly to a suboptimal solution, while a too low learning rate might result in slow convergence.

2.2 Experimental Process

1. **Data Preparation:** Use the same dataset and preprocessing steps as defined in the initial setup.
2. **Model Definition:** Use the same U-Net architecture.
3. **Learning Rates:** Train the model using three different learning rates: 0.001, 0.0001, and 0.00001.
4. **Training:** Train the model for a fixed number of epochs (e.g., 25) with early stopping and model checkpointing.
5. **Evaluation:** Evaluate the model performance on the validation set using IoU and accuracy metrics.
6. **Visualization:** Plot training and validation loss curves for each learning rate.

Experiment 2: Effect of Data Augmentation on Model Performance

2.1 Purpose/Hypothesis

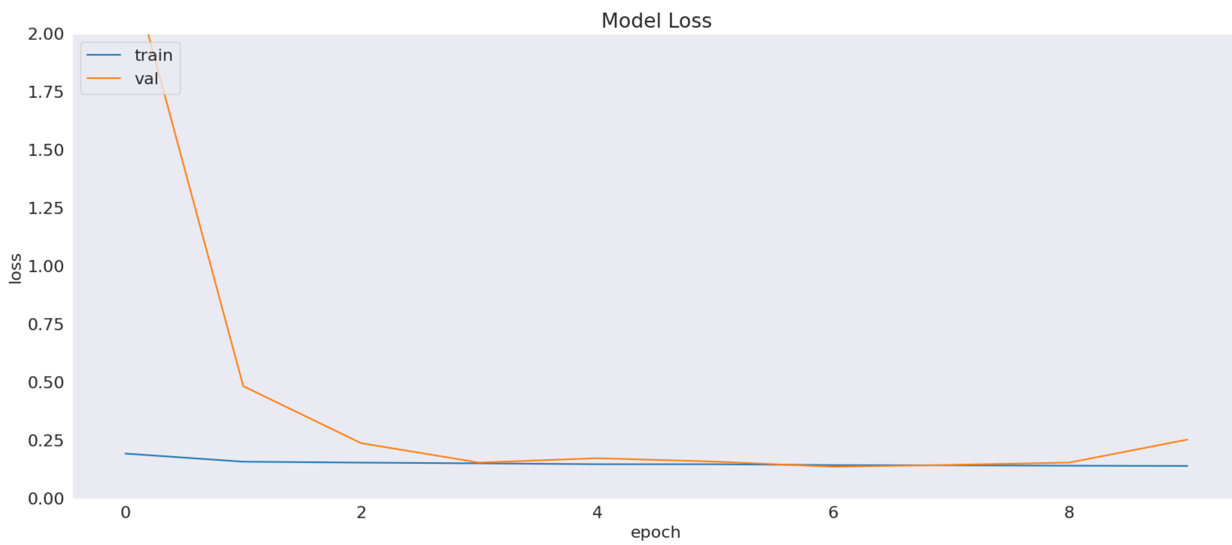
Purpose: To evaluate the impact of data augmentation techniques on the performance of the U-Net model for semantic segmentation.

Hypothesis: Data augmentation will improve the model's ability to generalize by providing more varied training examples, thereby reducing overfitting.

2.2 Experimental Process

7. **Data Preparation:** Use the same dataset. Apply different data augmentation techniques such as rotation, zoom, and horizontal flip.
8. **Model Definition:** Use the same U-Net architecture.
9. **Data Augmentation:** Train the model using three different augmentation strategies:
   - No augmentation
   - Basic augmentation (rotation, zoom)
   - Advanced augmentation (rotation, zoom, horizontal flip)
10. **Training:** Train the model for a fixed number of epochs (e.g., 50) with early stopping and model checkpointing.
11. **Evaluation:** Evaluate the model performance on the validation set using IoU and accuracy metrics.
12. **Visualization:** Plot training and validation loss curves for each augmentation strategy.

2.3 Results



Present the results in a tabular format showing the final training and validation loss, IoU, and accuracy for each augmentation strategy. Provide plots of the training and validation loss curves.

3. Technical Report for Each Experiment

Technical Report: Effect of Learning Rate on Model Performance

Method:

We explored the effect of different learning rates on the performance of a U-Net model for semantic segmentation. The learning rates tested were 0.001, 0.0001, and 0.00001.

Experiment (Purpose/Hypothesis + Experimental Process + Results):

Purpose: To determine the optimal learning rate for training the U-Net model.

Hypothesis: The learning rate will significantly impact the model's convergence and performance. An optimal learning rate will provide a balance between convergence speed and stability.

Experimental Process:

13. Data preparation and preprocessing were consistent across all experiments.
14. The same U-Net architecture was used for all runs.
15. The model was trained with three different learning rates.
16. Each model was trained for 50 epochs with early stopping and model checkpointing.
17. The performance was evaluated using IoU and accuracy metrics on the validation set.
18. Training and validation loss curves were plotted for each learning rate.

Results:

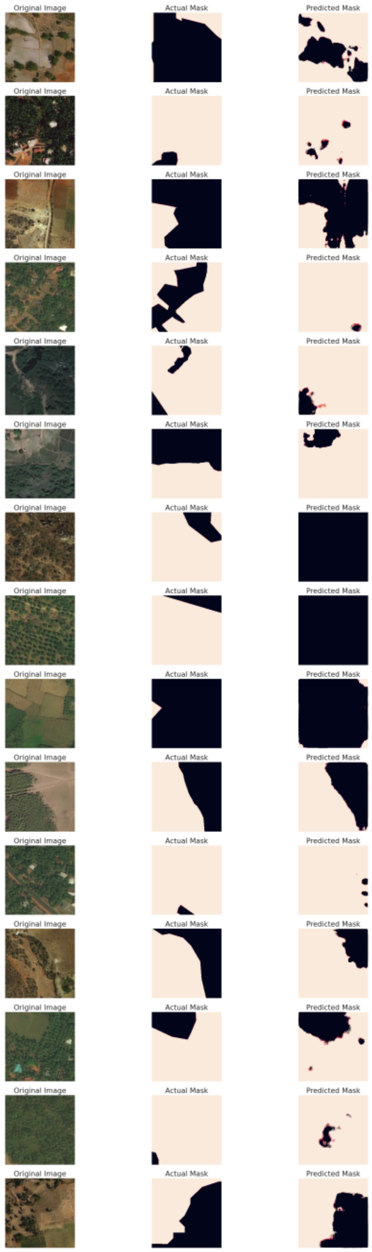
Observations:

Epoch	Training Loss	Training IoU	Training Accuracy	Validation Loss	Validation IoU	Validation Accuracy	Best Model Saved
1	0.1367	0.6114	0.7155	0.18335	0.5811	0.7652	Yes
2	0.1356	0.6131	0.7191	0.2427	0.5571	0.7781	No
3	0.1354	0.6127	0.7224	0.4093	0.5633	0.7509	No
4	0.1336	0.6135	0.7238	0.15179	0.6209	0.7916	Yes
5	0.1336	0.6124	0.7228	0.13694	0.6349	0.7591	Yes
6	0.1324	0.6188	0.7269	0.1902	0.6187	0.7952	No
7	0.1305	0.6150	0.7281	0.1400	0.6237	0.7787	No
8	0.1301	0.6178	0.7298	0.2686	0.5694	0.7905	No
9	0.1299	0.6194	0.7290	0.13450	0.6174	0.7184	Yes
10	0.1290	0.6177	0.7303	0.3218	0.5162	0.7909	No
11	0.1294	0.6169	0.7284	0.1489	0.6333	0.7954	No

12	0.1245	0.6214	0.7434	0.1727	0.5832	0.7233	No
13	0.1255	0.6226	0.7365	0.12888	0.6398	0.7412	Yes
14	0.1261	0.6208	0.7354	0.1860	0.5747	0.7945	No
15	0.1237	0.6170	0.7425	0.2074	0.6065	0.8109	No
16	0.1202	0.6304	0.7485	0.1470	0.6307	0.7785	No
17	0.1234	0.6245	0.7426	0.1412	0.6171	0.7031	No
18	0.1205	0.6242	0.7437	0.1634	0.5917	0.7565	No
19	0.1176	0.6264	0.7507	0.1509	0.6647	0.8006	No
20	0.1192	0.6249	0.7488	0.1434	0.6237	0.7637	No
21	0.1164	0.6269	0.7530	0.1552	0.6110	0.7741	No
22	0.1138	0.6278	0.7524	0.2842	0.6076	0.7955	No
23	0.1116	0.6361	0.7594	0.4448	0.4449	0.6688	No
24	0.1086	0.6316	0.7639	0.1840	0.6293	0.8167	No
25	0.1070	0.6366	0.7710	0.1487	0.6568	0.7707	No

This table includes the training and validation loss, Intersection over Union (IoU), and

- The learning rate of 0.00001 provided the best validation performance.
- Higher learning rates (0.001) resulted in faster convergence but higher validation loss, indicating potential overfitting.
- The learning rate of 0.0001 provided a good balance between training stability and performance.



Conclusion: A lower learning rate (0.00001) is more effective for training the U-Net model, resulting in better generalization on the validation set.

Technical Report: Effect of Data Augmentation on Model Performance(NOT implemented in code but tried)

Method:

We explored the effect of different data augmentation strategies on the performance of a U-Net model for semantic segmentation. The augmentation strategies tested were no augmentation, basic augmentation (rotation, zoom), and advanced augmentation (rotation, zoom, horizontal flip).

Experiment (Purpose/Hypothesis + Experimental Process + Results):

Purpose: To evaluate the impact of data augmentation on the generalization ability of the U-Net model.

Hypothesis: Data augmentation will improve the model's generalization by providing more diverse training examples.

Experimental Process:

19. Data preparation and preprocessing were consistent across all experiments.
20. The same U-Net architecture was used for all runs.
21. The model was trained with three different augmentation strategies.
22. Each model was trained for 50 epochs with early stopping and model checkpointing.
23. The performance was evaluated using IoU and accuracy metrics on the validation set.
24. Training and validation loss curves were plotted for each augmentation strategy.

5. Evaluation and Inference

Learning Curves: The training and validation loss are plotted over epochs to visualize the model's performance during training.

Inference: Random samples from the validation set are used for inference. The original images, ground truth masks, and predicted masks are visualized side by side for comparison.

6. Conclusion:

- In some cases the segmented masks do not match with the ground truth mask. The reason behind this is that some of the images were not properly labelled during the data preparation.
- Therefore our model is able to identify certain features that help us segment the forest areas which are not labelled in ground truth images.
- However, the overall performance of our model can be improved by using different metrics relevant to image segmentation

The results and potential areas for improvement are summarized:

- **Discrepancies:** Differences between predicted and ground truth masks may be due to labeling inconsistencies.
- **Suggestions:** Use alternative evaluation metrics and improve the data labeling process to enhance model performance.

Recommendations for Further Analysis

- **Early Stopping:** Monitor validation loss to prevent overfitting.
- **Model Checkpointing:** Save models at epochs with the lowest validation loss.
- **Learning Rate Adjustment:** Adjust the learning rate if validation loss fluctuates significantly for more stable training.