Finetuning U-Net for Semantic Segmentation of Windows in Images of Buildings

1 Problem Definition

Semantic segmentation is a highly important and useful task that involves classifying pixels in an image. This project investigates the extent to which fine-tuning a pre-trained U-Net segmentation network can solve the problem of segmenting windows in an image, given limited training data. The model is expected to, when given an RGB input image of a building with windows, produce a binary segmentation mask of it. To achieve this with accuracy and efficiency, a pre-trained image segmentation model was fine-tuned. Loss functions, regularisation methods, and data augmentations were compared and tested to extract the best performance from the model. The final window segmentation model was trained using only 4 epochs, and succeeds in segmenting a range of online images, evidenced by Figures 7-20.

2 Method

The model is based on the SMP (Segmentation Models Pytorch) U-Net model, which is pre-trained on ImageNet with a ResNet34 encoder [3]. A supervised learning approach was used. Training was chosen to be end-to-end with all weights undergoing optimisation, producing superior results [1]. A training set of 198 images and corresponding binary ground truth masks was created. Masks were produced by manually segmenting the training images using the SAM (Segment Anything Model) from Meta AI [4]. The training set consists of the original JCSMR image resized to 768 by 768, along with images produced by ArchitectureRealMix v1.1 (based on Stable Diffusion) when conditioned on JCSMR and additional prompts [5]. The images were generated in 30 steps, on a dpm++ 2m gpu sampler, karras scheduler, 0.85 denoising, and a resolution of 768x768.

Different levels of data augmentation were tested: no augmentation, only random crops, and then a mixture of random crops, flips, rotations, and Gaussian noise. The crop size taken was 400 by 400, to ensure less similarity in each crop while capturing spatial variation in the image. I used the albumentations, SMP, and timm libraries for their support in synchronising geometric transforms between images and masks; otherwise, the data-to-label correspondence is lost. During training, a variety of loss functions were tested to find the most appropriate for the task: Tversky loss, with $\alpha = 0.3, \beta = 0.7$, balanced Tversky, with $\alpha = 0.5, \beta = 0.5$, Focal loss, Binary Cross Entropy (BCE), and a weighted combination of Focal and Tversky (Focal-Tversky) losses. The use of ℓ^2-regularisation in mitigating overfitting was investigated by using a weight-decay parameter of 10^{-6}. This is equivalent to penalising the model as follows: $\mathcal{L}' = \mathcal{L} + \lambda||w||^2$, $\lambda = 10^{-6}$ for all weights w in the network.

A 13-image-mask validation set of images of various buildings sourced online, including others from the ANU, was compiled to generate loss curves for benchmarking on Tensorboard. Validation loss curves were used to train the model without overfitting.

3 Results

In investigating data augmentation, a BCE loss was used to train the model. Evidently from the validation loss curves in Figure 1 and Figure 2 the model performs best when trained on well-augmented data that includes random crops, flips, rotations, and noise. A data set with only random crops reaches a suboptimal solution (see Figure 1), with a loss above 0.2. As seen in Figure 2, the best number of augmentations per image is around 15, corresponding to an optimal validation BCE loss of 0.125 at 8 epochs. Rapid increases in validation loss at 20 epochs in Figure 2 indicates overfitting, suggesting that 15 augmentations per image is optimal. For further

investigation of loss functions and regularisation, these optimal augmentations were maintained throughout training.

ℓ^2-regularisation was investigated by training the model using the 10^{-6} weight decay provided by SMP. Comparison of Figure 2 and Figure 3 shows that regularisation was counterproductive, preventing the model learning from images to the required extent - the minimum validation loss of 0.16 is higher than before. Visually, in the predicted masks, fewer windows were shaded (see Figure 9), and the model predicted more false negatives. For this task, regularisation is unnecessary because of the small data set and minimal training required to fine-tune; overfitting is not the largest concern here.

The choice of loss function significantly affected the model performance. Tversky loss with $\alpha = 0.3$ and $\beta = 0.7$ which penalised false negatives was tried. However, it did the worst, classifying far too many regions as windows. Tversky with $\alpha = 0.5, \beta = 0.5$ achieved a particularly strong and accurate segmentation mask, but in Figure 12 it does not handle thin separation between windows properly. Focal loss successfully handled thin separation, but it also led to classifying too many pixels as windows, as seen in Figure 18, and weakly segmenting regions. To find a balance, a combination of Focal and Tversky Losses was used, with the loss function $\mathcal{L} = 0.5\text{Focal}(\text{prediction}, \text{target}) + 0.5\text{Tversky}(\text{prediction}, \text{target})$. The loss function that produced the best results was a balanced combination of Focal and Tversky (Focal-Tversky) losses. In particular, the model trained on this was able to segment difficult regions in buildings with unique architecture, as evidenced by different views of JCSMR. This is because Focal Loss is great at focusing on hard examples, and Tversky loss produces strong segmentations [2]. Combining both balances their limitations. The minimum validation loss of the Focal-Tversky was approximately 0.18 BCE loss (see Figure 4), which was less than most previously tested models. Although the BCE losses were lower for the model trained using BCE, this is natural because the model was trained and validated on BCE. Visually, Focal-Tversky produced stronger segmentations and was better at handling thin boundaries. In terms of efficiency, Focal-Tversky loss took only around 4 epochs to converge, but BCE converged after 8 epochs (see Figure 4). To compare visuals, see Figure 10, Figure 16, and Figure 19.

4 Reflection

Although segmenting windows could be useful, especially for surveillance purposes, it can be particularly damaging. Due to the transparency of windows (in general), a model capable of detection that automates locating windows in different buildings could be modified for stronger surveillance or military applications. For example, a sniper would greatly benefit from knowing where precisely the windows in a building are located. Consent is another concern that arises while sourcing training data - real images of buildings need to be ethically sourced with consent from their owners. Without consent, sourcing this data is unethical.

5 Conclusion

The proposed model has been implemented and trained, with a Focal-Tversky loss function and data augmentation, trainable with only 4 epochs. It would be interesting to further explore how well models can be fine-tuned using only scarce real data and AI-generated data for augmentation, and the performance difference compared to only using real training data. Training segmentation models on scarce or minimal data is a wide area of research that has many applications. In practice, even fully supervised learning is not always possible, as it requires significant manual labour. A semi-supervised approach for this problem on a larger AI-generated dataset would make an interesting subject for further exploration.

References

[1] D. Cheng and E. Y. Lam, "Transfer Learning U-Net Deep Learning for Lung Ultrasound Segmentation," arXiv:2110.02196 [cs, eess], Oct. 2021, Available: https://arxiv.org/abs/2110.02196

[2] R. Azad et al., "Loss Functions in the Era of Semantic Segmentation: A Survey and Outlook," arXiv.org, Dec. 08, 2023. https://arxiv.org/abs/2312.05391

[3] SMP GitHub Repository. *segmentation_models.pytorch*. Available at: `https://github.com/qubvel-org/segmentation_models.pytorch`

[4] Segment Anything Model Repository. `https://github.com/facebookresearch/segment-anything`

[5] Stable Diffusion. `https://openart.ai/workflows/home?utm_source=google%2Cgoogle&utm_medium=pmax%2Cpmax&utm_campaign=Performance_Max_Science_Education%2C21325416878&utm_term=`

[6] Exterior of Hanna Neumann. `https://x.com/markstickells/status/1385187309864177665/photo/1`

[7] Full view of Hanna Neumann. `https://www.sutersarch.com/project/anu-hanna-neumann-building`

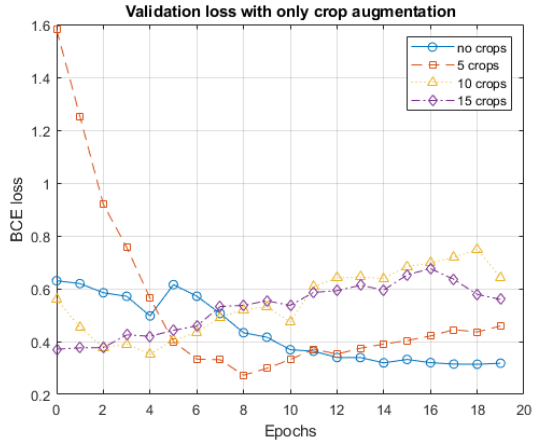

Figure 1: Validation: only crop augmentation

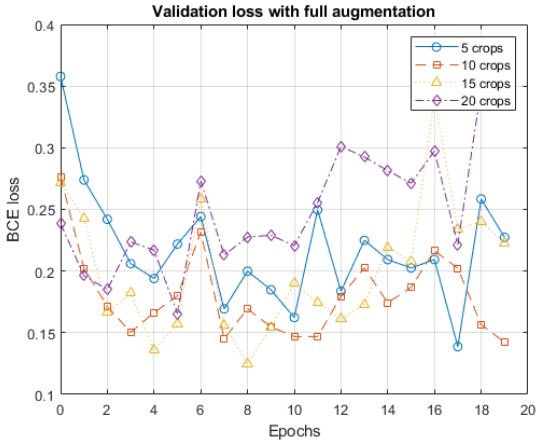

Figure 2: Validation: full augmentation

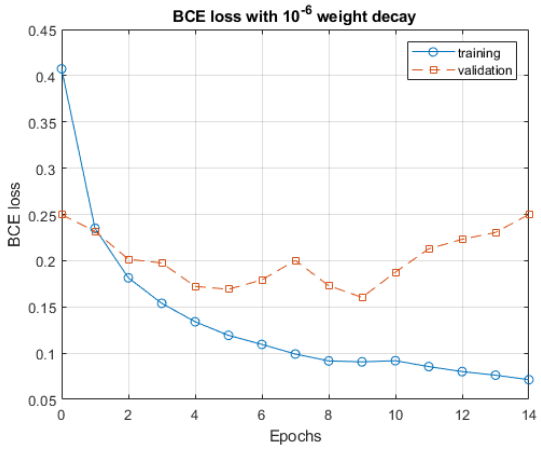

Figure 3: l^2 regularisation

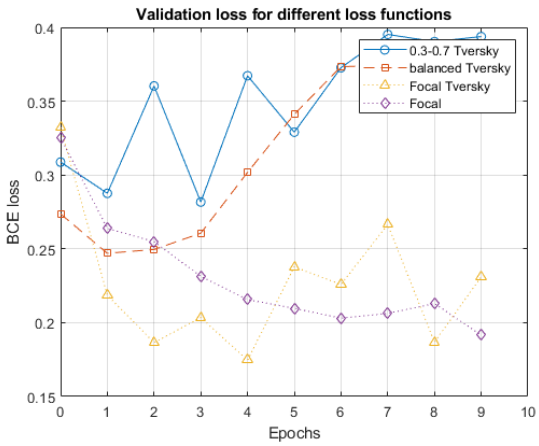

Figure 4: Using different loss functions

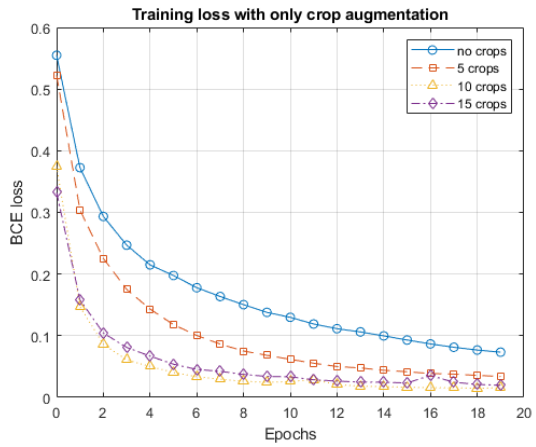

Figure 5: Training: only crop augmentation

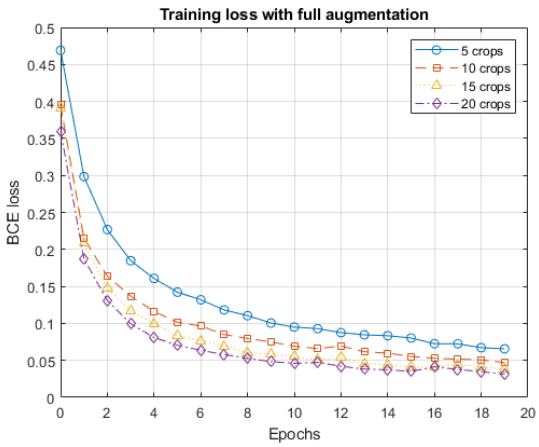

Figure 6: Training: full augmentation

Figure 7: Exterior of Hanna Neumann [6]

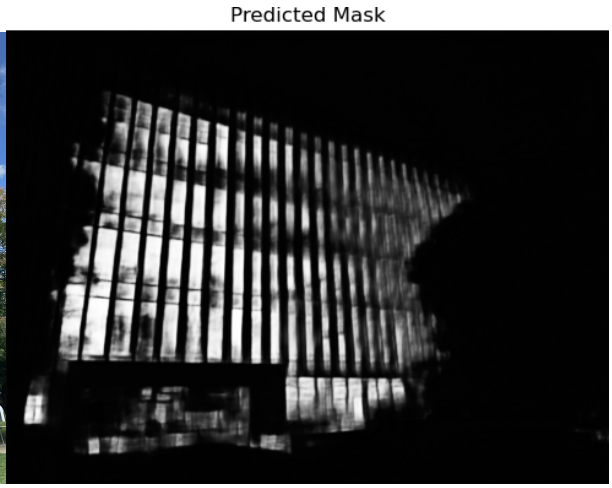

Figure 8: BCE

Figure 9: BCE with regularisation

Figure 10: Focal-Tversky

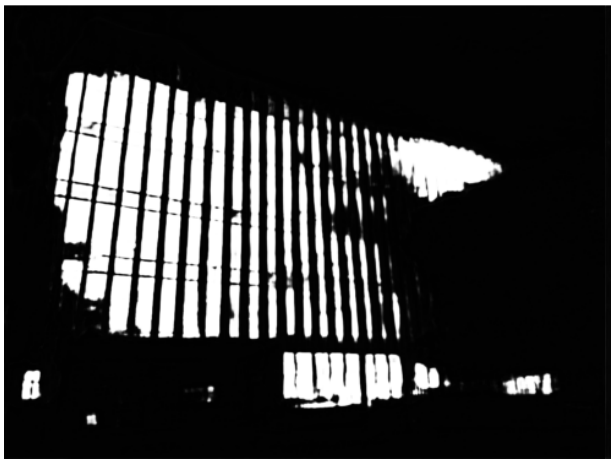

Figure 11: Tversky, $\alpha = 0.3, \beta = 0.7$

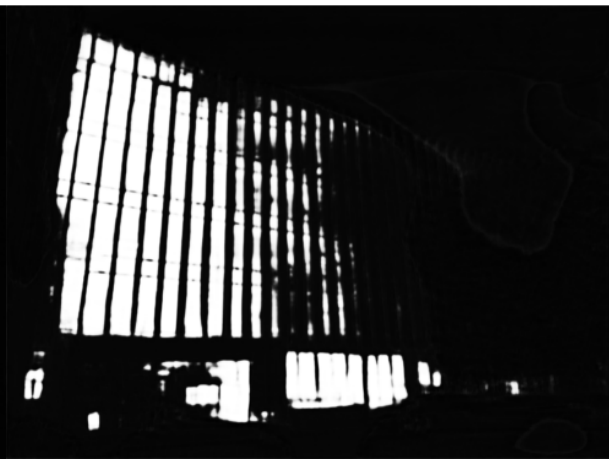

Figure 12: Tversky, $\alpha = 0.5, \beta = 0.5$

Figure 13: Full view of Hanna Neumann [7]

Figure 14: BCE

Figure 15: Focal

Figure 16: Focal-Tversky

Figure 17: A view of JCSMR [8]

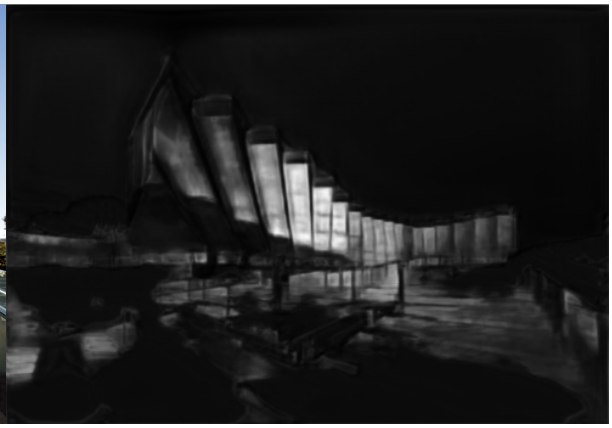

Figure 18: Focal

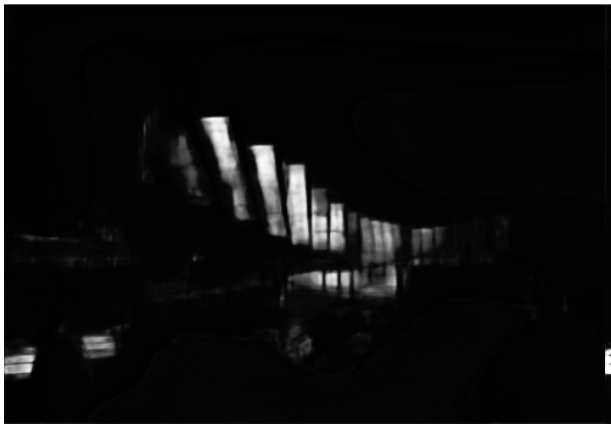

Figure 19: Focal-Tversky

Figure 20: Tversky, $\alpha = 0.5, \beta = 0.5$