CS-433 Machine Learning Project 2: Road Segmentation with CBAM2U-Net

Jerome Wang, Weng Pei He
EPFL, Switzerland

***Abstract*—U-Net has been widely regarded the go-to model for image segmentation, which speaks to its efficacy in segmentation tasks. However, the semantic gaps which exist in different components of U-Net results in wasted potential. Thus, this paper proposes the use of Convolutional Block Attention Mechanism to bridge such semantic gaps, which is proven to improve performance on "Hard" samples.**

I. Introduction

A. Context

With copious amounts of Satellite imaging data online, accurate processing of such images become ever more important. Thus, image segmentation now plays an even more important role in utilizing the wealth of such data.

B. Related Work

U-Net [1] is a well established encoder-decoder model architecture commonly used for image segmentation. However, it suffers from semantic gaps - both between successive encoder layers [2] as well as between encoder and decoder [3]. Thus, this paper explored the Convolutional Block Attention Module (CBAM) [4], a novel Computer Vision specific attention-based mechanism to better focus the model attention on more relevant areas.

C. Our Contributions

Inspired by Woo et. al. [4] who placed CBAM modules after convolutional blocks in ResNet, this paper similarly proposes for CBAM modules to be placed after each convolutional block in both the encoder and decoder sections of the U-Net. Thus, we name our proposed model CBAM2U-Net. Typical deep learning best practices such as randomly augmenting the training data and the setting of a reasonable tile size were also adopted.

II. Models and Methods

A. Exploratory Data Analysis and Preprocessing

Exploratory Data Analysis: Firstly, preliminary exploratory data analysis was performed on the dataset containing 100 training images and 50 test images, where it was observed to depict a variety of roads: main roads, highways, residential roads or carparks. A non-trivial amount of occlusions is present in several of the images (i.e. cars, road markings, trees), which is hypothesized to present a fair amount of challenge for vanilla U-Nets where there is no attention mechanism to focus the training.

Fairly significant pixel imbalance was also observed, with the ROI class (roads) only making up approximately 20% of the images.

Binarizing: Next, we applied a binary threshold mask to our ground truth images for training using opencv_python, as the given mask images are in RGB format whereas the model mandates a binary image format. Using the threshold of 1, all pixel values above that threshold was reassigned to 1. Additionally, the number of image channels was reduced from 3 to 2. Thus, the resultant masks ready for input to the model is of shape (400, 2) with values $\{0, 1\}$.

Tiling: Fixed-window tiling and padding were also performed on test images since test images were larger in size (512, 512) as compared to the train images (400, 400). This input size was chosen to avoid losing any contextual information from our training images, which could occur with a smaller tile size. For example, a road may be split along the middle between 2 tiles, or a T-shape intersection may be cut into half. Conversely, if a tile size larger than the training image sizes were to be used, parameters will be needlessly wasted as the images would have to be (center-)padded, which will see an increase in the number of model parameters to be trained without seeing a visible improvement of model performance since the model would be training on trivial pixels (padded pixels).

Augmentation: As the last step in the image preprocessing pipeline, the training images (and ground truth images) are randomly altered before they are fed to the model, which helps to make the model more robust, which can improve the model's generalisation and performance on unseen data.

Such random augmentation was implemented with keras' pre-processing layers, where the random rotation, random brightness and random contrast layers were used to transform the dataset. Random rotation is posited to help the model to generalize better and reduce over fitting. For example, it will teach the model that roads are not necessarily perpendicular to the image frames, while random brightness and random contrast are posited to help with generalising road conditions across different times of day, as well as different shades of the road and its surroundings.

B. Model Architectures

1) U-Net: The U-Net is a symmetrical network comprising of both the encoder and decoder halves, with each layer of each half being a mirror image of the other.

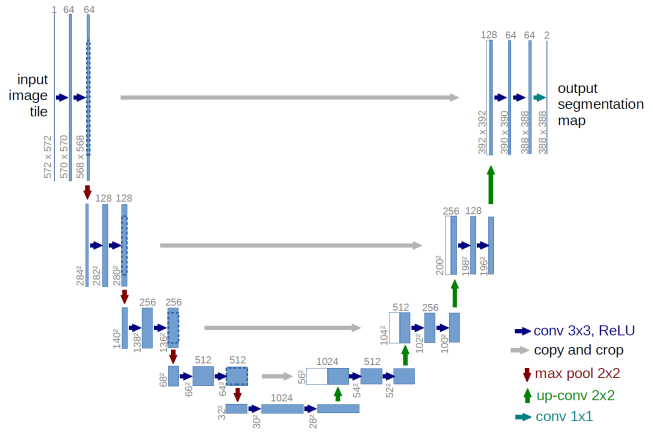


Figure 1: U-Net architecture [1]

Figure 1 describes the general structure of a U-Net. It consists of two halves: the encoder, which learns the features from an input image, as well as the decoder, which generates the output segment maps. Each blue rectangle refers to a multi-channel feature map, with the number of channels at the top and the size (in pixels) at the left.

The encoder behaves almost like a standard encoder in a convolutional neural network, and both convolutional and pooling layers are used. One notable difference is the use of double convolution: 2 stacked convolutional layers before pooling. These layers use a filter size of 3x3 and stride 1, as well as valid padding. A ReLU activation layer is included after convolution for non-linearity. For pooling, a 2x2 max pooling function is used, with a stride of 2. As with classical encoders, the encoder half of the U-Net extracts context at each level of spatial dimension by down-sampling feature maps with each successive encoder layer. However, such down-sampling of feature maps results in a semantic gap between the ends of the encoder as feature resolution is lost.

The decoder is where the U-Net differs from classical CNNs. In the place of a fully connected neural network, the decoder uses an assortment of normal convolutions, up-convolutions, skip connections to generate the segmented output map. The normal convolutions used are the same double convolution layers from the encoder. Up-convolution acts as a sort of 'reverse-pooling': where 1 dimensional feature vectors are mapped to 2x2 pixel output using learned weights.

At each encoder layer, a skip-connection is formed to the decoder-layer, which ostensibly helps to preserve lower-level features that may otherwise be lost amidst successive convolution and feature fusion [1]. However, the usefulness of these direct skip-connections remains controversial, since a non-trivial semantic gap exists between the encoder and decoder halves of the U-Net. There is a disconnect between the features extracted by the encoder at each network depth - while deeper layers are proficient at semantically understanding the image, they may not fare well in clearly defining the edges of each segment [5]. This has caused the U-Net's performance on certain datasets to be worse off than if skip connections weren't used [3].

2) CBAM: Convolutional Block Attention Module: A potential solution to these semantic gaps will be to use a discriminatory skip connection by means of an attention-based mechanism. The Convolutional Block Attention Mechanism (CBAM), as proposed by Woo et al [4], achieves this. CBAM focuses on both the spatial and channel aspect of images, highlighting important features and downplaying less important ones through weighing.

Channel attention focuses on the semantic context of an image.

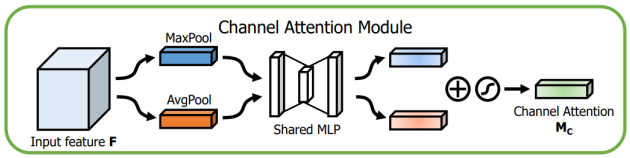


Figure 2: CBAM's Channel Attention Module [4]

The channel module can be represented with the following equation:

$$M_c = \sigma(MLP(AvgPool(F)) + MLP(MaxPool(F)))$$

Where M_c is the channel attention map, MLP being a Multi-Layer Perceptron with ReLU activation, σ being the sigmoid activation function and F being the input feature map.

On the other hand, Spatial Attention focuses on the localization of objects (spatial context) in the image. The spatial

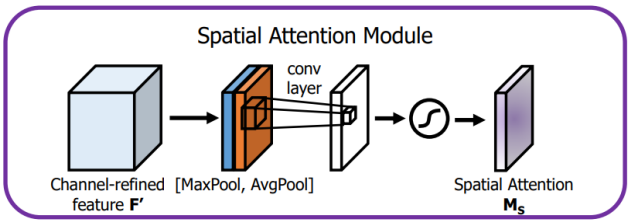


Figure 3: CBAM's Spatial Attention Module [4]

attention module can be represented with the following equation:

$$M_s = \sigma(f^{7x7}([AvgPool(F); MaxPool(F)]))$$

Where M_s is the spatial attention map, σ the sigmoid activation function, f^{7x7} the convolution with 7x7 filter and F being the input feature map.

The CBAM module is completed when the channel module is arranged first in a sequence with the spatial module with an identity mapping between the input feature and the channel and spatial refined feature, as depicted below.

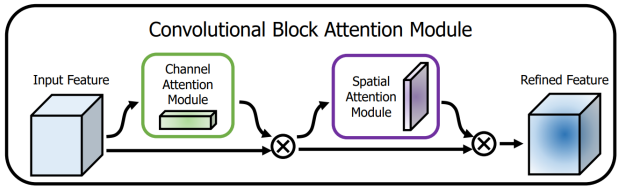


Figure 4: CBAM architecture [4]

C. Model Implementation

Since CBAM has proven to help improve the performance of ResNets when placed after each convolutional block [4], it is evident that the suppression of less relevant features relating to both channel and spatial aspects helps the model it is attached to to effectively discriminate more important features to train on. This will be especially beneficial in the context of bridging the semantic gap between 1) encoder layers and 2) encoder and decoder. Thus, in the encoder, CBAM will be placed right after the convolutional block and right before the skip-connection, which will benefit both the encoder and decoder layers since it is posited to help address the semantic gap between encoder layers, as well as the semantic gap between encoder and decoders at all layers (where the skip-connections exist). Similarly, CBAM will be placed right after the convolutional block in the decoder layers, as well as after the bottleneck in between encoder and decoder layers.

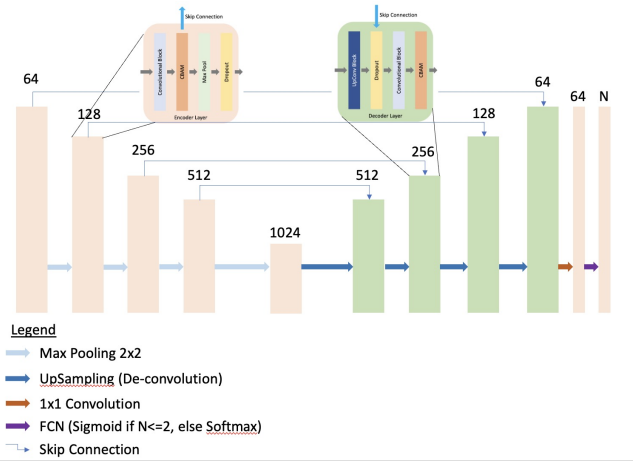


Figure 5: Proposed Encoder Layer Structure

D. Training

Validation: To prevent the model from over-fitting, a train-validation split was performed. For this purpose, we arbitrarily chose the last 10 images of the training set to be the validation set. While K-fold cross-validation would indeed be a preferable alternative, it has proven too costly to implement, since it would require us to train our model multiple times.

Early stopping : To prevent over-fitting, we made use of early stopping callbacks with Keras. This would stop a model's training after a number of epochs with no significant improvement made on a validation dataset. The number of epochs is the patience value, which we initially set at 50. Empirical tests has shown 50 to be an effective threshold for the bias-variance tradeoff: high enough to learn all the parameters of our model, and low enough to avoid over-fitting. This was used to identify a suitable epoch value to be used across the training of both models, after which it was removed in order to maintain parity between their training cycles.

Metrics and Loss: We use several metrics to measure the performance of our model, as well as to train it.

$$IoU = \frac{TP}{TP + FP + FN}$$

Intersection over Union, also known as Jaccard index, represents the area of the intersection of the prediction and the ground truth over the union of the prediction and ground truth. The intersection area is easily calculated by pixel counting.

$$Dice = \frac{2TP}{2TP + FP + FN}$$

The Dice Coefficient, also known as the Sørensen-Dice index or the F-1 Score is another method for measuring the performance of binary segmentation tasks. It is the harmonic mean of precision and recall.

Both the IoU and Dice coefficient provides a good measure of performance of the model, since they are sensitive to both precision and recall. We choose the Dice coefficient for our loss function since it is slightly more sensitive to small changes in the overlap between prediction and ground truth.

$$L_{dice} = 1 - Dice$$

The dice loss is given by simply subtracting the Dice coefficient value from 1. In our implementation, we weigh the dice loss with the inverse of the class frequency in order to combat class imbalance when training the model, and encourage the model to give more importance to the underrepresented road class.

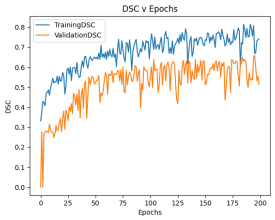
Training process: Model training is completed using Adam Optimizer, with an initial learning rate of 1e-4(0.0001).

III. Results

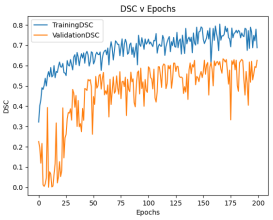
Averaged metrics for baseline and proposed models are as follows:

Model	IoU	Dice Coefficient
U-Net	0.67 ± 0.09	0.52 ± 0.09
CBAM2U-Net	0.63 ± 0.11	0.47 ± 0.12

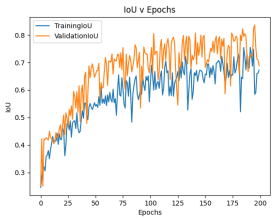
Table I: Model Performance on validation set



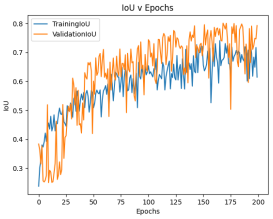
(a) Dice Coefficient across Epochs for U-Net



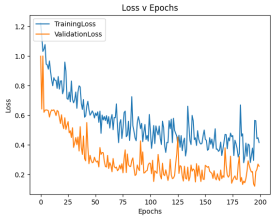
(b) Dice Coefficient across Epochs for CBAM2U-Net



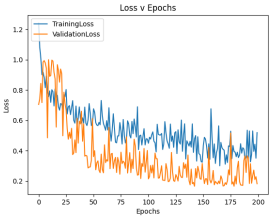
(a) IoU across Epochs for U-Net



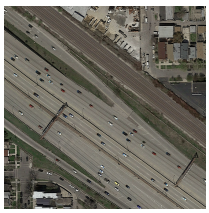
(b) IoU across Epochs for CBAM2U-Net



(a) Dice Loss across Epochs for U-Net

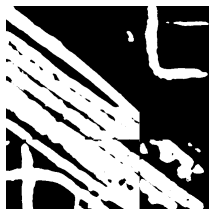


(b) Dice Loss across Epochs for CBAM2U-Net



(a) Source Image from Test set

(b) Prediction by U-Net



(c) Prediction by CBAM2U-Net

Figure 9: Case Study of model performance

IV. Discussion

Quantitatively, both U-Net and CBAM2U-Net has similar performances as the spread of both's validation results are within similar ranges. Consequently, both the vanilla U-Net as well as the CBAM2U-Net produced largely comparable images. However, CBAM2U-Net faired better on "Hard" samples - samples which contain roads clustered tightly together. Results from U-Net had poorer delineation between road instances as compared to those from CBAM2U-Net, which points to the effectiveness of the CBAM module at addressing the afore-mentioned semantic gaps that U-Net suffers from.

By providing discriminatory power to skip-connections, CBAM has mitigated the semantic gap between Encoder and Decoder, as encoder layer features are no longer forwarded wholesale to decoder layers.

Additionally, the semantic gap between successive encoder layers are also mitigated since the most important channel and spatial features are emphasized by the CBAM module. In the vanilla U-Net, even though deeper layers of the encoder may be proficient in semantically understanding the image, it may not fare well in clearly defining the edges of each segment [5]. Such is a trade-off of convolution, as features gets compressed (projected into a smaller dimensional space) to lower computational cost. This is not ideal, as both low- and high-level features are interdependent for an accurate segmentation [2]. In contrast, for a U-Net encoder with CBAM, the negative effects of such a trade-off is mitigated since the most important features are retained throughout successively convolutions.

However, CBAM was expected to have a better performance than what was recorded during the experiment. This could be down to the small batch size used (which was limited by the computational resources we had access to), since CBAM relies on global pooling (both average and max) to attend to various features in the image. With a smaller batch size, lesser features are available in the global feature space, which does not allow CBAM to play out to its potential.

Additionally, both models had relatively volatile losses, which could be down to the optimizer use - as models with Adam tends to be more aggressive with its learning than those with Stochastic Gradient Descent. This could have resulted in the U-Net model stopping training at a advantageous training step, whereas the opposite may have occured for CBAM2U-Net.

Though SGD takes smaller learning steps (thus converging at a slower rate) and hence offers for more accurate fine-tuning using TensorFlow's in-built ReduceLROnPlateau function, empirical tests has shown that its qualitative performance was worse than that of Adam's, with less clearly-defined segments and more mis-predictions.

V. Summary

To conclude, though U-Net provides for a solid baseline for image segmentation, due to the semantic gaps that exist in the architecture, it leaves room to be desired when it comes to "Hard" samples. CBAM, with its attention mechanism, can help bridge those gaps, providing better performance for "Hard" samples. In the experiments carried out, the CBAM results are quantitatively similar to that of the Vanilla U-Net, which could have been due to the small batch size and volatile losses.

VI. Ethical Risks

One risk inherent with this project of road segmentation is fariness.

The data set could be subject to representation bias: it seems to contain mostly asphalt roads set in urban environments, during sunny weather. As such, models trained on this dataset might fail to account for differing lighting, road type, and weather conditions. This would disproportionately impact certain communities or regions. For example, roads made with mud or dirt might not be recognised by the model. It could also fail to recognise roads in underdeveloped areas, where there is poor lighting at night, or areas with poor weather and little sun. With this risk, the stakeholders involved are indirectly affected.

Road segmentation has many uses. It can be used to guide an urban development plan or optimise routes for unmanned vehicles. In these applications, the mislabeling of roads due to bias can have a significant impact. For example, it is not too egregious to imagine that city planners may be deterred from constructing public facilities such as libraries, water treatment plants or power substations in poorer neighbourhoods due to an imagined lack of accessibility for construction vehicles, brought about by the fact that a road segmentation model shows that there are few usable roads in this area. In another example, the mistaken segmentation of roads in a foggy area results in autonomous vehicles in the area being redirected to another road nearby, thus causing a traffic jam in the area.

The likelihood of occurrence of this risk would depend on the quality of input data. If a model is trained on widely varied data, the likelihood of a mislabel occurring would be rather low. The inverse is also true. Given the one-dimensional nature of our current dataset, the likelihood of mislabelling can be deemed to be relatively high. Thus, when weighing the potential impact of the risk and its likelihood, we come to the conclusion that this is definitely a non-negligible risk, and should be addressed.

To address this risk, we make use of image augmentation. By altering the brightness and contrast of our training images, we can simulate different lighting and weather conditions. While this would not eliminate the risk completely, it does help to reduce the likelihood of mislabelling.

It is worthy to note that predictions from both U-Net and CBAM2U-Net contain tiling artifacts - there is an obvious disjoint in lines at where the top-left tile ends, which is consistent throughout the dataset. A possible way to mitigate this would be to use the sliding-window tiling method, rather than a fixed-window tiling method (as was used in this paper).

References

[1] O. Ronneberger, P.Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, ser. LNCS, vol. 9351. Springer, 2015, pp. 234–241, (available on arXiv:1505.04597 [cs.CV]). [Online]. Available: http://lmb.informatik.uni-freiburg.de/Publications/2015/RFB15a

[2] Y. Pang, Y. Li, J. Shen, and L. Shao, "Towards bridging semantic gap to improve semantic segmentation," in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019, pp. 4229–4238.

[3] S. Wang, V. K. Singh, E. Cheah, X. Wang, Q. Li, S.-H. Chou, C. D. Lehman, V. Kumar, and A. E. Samir, "Stacked dilated convolutions and asymmetric architecture for u-net-based medical image segmentation," *Computers in Biology and Medicine*, vol. 148, p. 105891, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0010482522006308

[4] S. Woo, J. Park, J.-Y. Lee, and I. S. Kweon, "Cbam: Convolutional block attention module," 2018.

[5] Z. Zhang, Q. Liu, and Y. Wang, "Road extraction by deep residual u-net," *IEEE Geoscience and Remote Sensing Letters*, vol. 15, no. 5, p. 749–753, May 2018. [Online]. Available: http://dx.doi.org/10.1109/LGRS.2018.2802944

Appendix

Hyperparameter	Value
Training Epochs	200
Batch Size	8
Dropout Ratio	0.3
Tile Size	400
Optimizer	Adam (1e-4)

Table II: A summary of the hyper-parameters used in training