PatchRefineNet: Improving Binary Segmentation by Incorporating Signals from Optimal Patch-wise Binarization

Savinay Nagendra Daniel Kifer
sxn265@psu.edu dkifer@cse.psu.edu
Department of Computer Science
The Pennsylvania State University
University Park

Abstract

The purpose of binary segmentation models is to determine which pixels belong to an object of interest (e.g., which pixels in an image are part of roads). The models assign a logit score (i.e., probability) to each pixel and these are converted into predictions by thresholding (i.e., each pixel with logit score $\geq \tau$ is predicted to be part of a road). However, a common phenomenon in current and former state-of-the-art segmentation models is spatial bias – in some patches, the logit scores are consistently biased upwards and in others they are consistently biased downwards. These biases cause false positives and false negatives in the final predictions. In this paper, we propose PatchRefineNet (PRN), a small network that sits on top of a base segmentation model and learns to correct its patch-specific biases. Across a wide variety of base models, PRN consistently helps them improve mIoU by 2-3%. One of the key ideas behind PRN is the addition of a novel supervision signal during training. Given the logit scores produced by the base segmentation model, each pixel is given a pseudo-label that is obtained by optimally thresholding the logit scores in each image patch. Incorporating these pseudo-labels into the loss function of PRN helps correct systematic biases and reduce false positives/negatives. Although we mainly focus on binary segmentation, we also show how PRN can be extended to saliency detection and few-shot segmentation. We also discuss how the ideas can be extended to multiclass segmentation. Source code is available at https://github.com/savinay95n/PatchRefineNet.

1. Introduction

Binary segmentation [12, 27, 32, 57] is the task of identifying which pixels in an image belong to objects of interest. Examples include identifying roads in satellite images [1, 71, 72] and polyps in medical images [27, 52, 73].

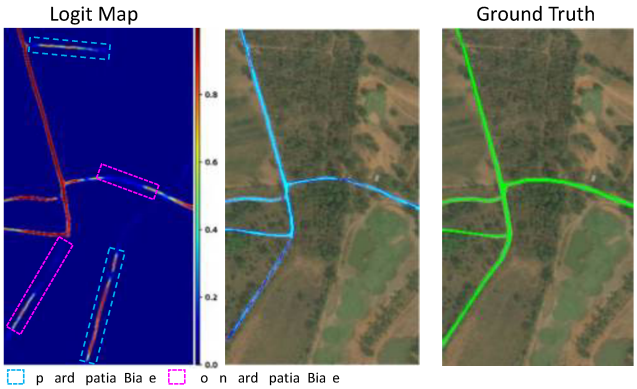

Figure 1. **Spatial bias example in an image from DeepGlobe [12].** Column 1: Logit map produced by CoANet [41]. Regions with significant downward and upward spatial biases are highlighted by boxes with the corresponding colors. Column 2: Debiased logit map produced by PRN. Column 3: Ground Truth.

Neural networks that are trained to perform binary segmentation typically output something called a *logit score* for each pixel – a number between 0 and 1 indicating the likelihood that this pixel belongs to the object of interest. The logit scores for all the pixels are collectively referred to as a *logit map*. The logit maps are converted into a final predictions through *binarization* –picking a threshold τ and setting a pixel's prediction to 1 if the corresponding logit is $\geq \tau$ and 0 otherwise.

Despite steady improvement in network architecture for binary segmentation models [2, 3, 39, 54], logit maps from former and current state-of-the-art networks exhibit spatial biases that limit the accuracy of the resulting binarized predictions. As an example, consider Fig. 1. The first column shows the logit map produced by CoANet [41], a high-performing road segmentation network, from an image from the DeepGlobe dataset [12] (the ground truth is shown in column 3). In the first column, the regions marked by

pink boxes represent image patches with significant downward bias in their logit scores. In these patches, the pixels that actually belong to roads have an average logit score of ≈ 0.2. Meanwhile, the cyan boxes represent image patches with significant upward bias in their logit scores. The lines shown inside those boxes have an average logit score of ≈ 0.7. Having significant amounts of *non-road* pixels with higher logit scores than actual road pixels is problematic – binarization will produce final predictions with many false negatives in the pink boxes and false positives in the cyan boxes. This type of spatial bias in logit maps is not specific to CoANet – it is a consistent trend for all segmentation networks we have tried. Meanwhile, the second column shows how our proposed PatchRefineNet (PRN) has removed the spatial biases.

Clearly, to handle spatial biases, the logit maps in different image patches should be handled differently (instead of being binarized in the same exact way). One naive approach is to allow each image patch to have its own threshold, and to have a neural network trained to predict what that patch-specific threshold should be (e.g., if it believes that logits are biased upward in an image patch, it can set a higher threshold for that patch). However, such an approach has an important shortcoming – it is too rigid. Even inside an image patch, there could be spatial variation in the bias. For example, in an image patch that is generally biased upwards, there will be many clusters of pixels with upwardly-biased logit scores, but there can still be clusters with downward biases or almost no biases. Binarizing such a patch with a single threshold can often result in clusters of false negatives/positives.

To address this problem, we propose **PatchRefineNet (PRN)**. One takes any segmentation network as a base and puts PRN on top of it (the input to PRN is the logit map produced by the base network). PRN learns the spatial biases of the base network and then adjusts the logit score of each pixel to compensate. PRN uses two learning signals during training. The first is the ground truth labeling of each pixel. The second is a novel learning signal from a set of "pseudo-labels" designed as follows: (1) for each image patch in a training image, one first finds an optimal threshold for binarizing that patch; (2) then one uses these patch-specific thresholds to binarize each patch. The resulting binarization of each pixel is the pseudo-label for that pixel. Intuitively, these pseudo-labels train PRN to detect the overall bias in a patch, while the ground truth learning signal trains PRN to detect the exceptions (e.g., clusters of pixels with a downward bias inside a patch that is generally upward-biased).

In order to learn about spatial biases in the base network, PRN splits an input logit map into k disjoint patches. There is a global branch that processes the entire logit map, which helps PRN understand the relationships between patches. There is also a local branch that processes individual patches (to learn about local properties/biases in a patch). Both branches produce logit maps which are then averaged (resulting in the "final" logit map) and then thresholded at 0.5 (for final binarized predictions).

Why don't existing networks automatically correct their own biases by training with the ground truth? We conjecture this is because in their training, the loss at a pixel-only level depends on the label and prediction for the pixel, hence the networks are not very good at noticing general trends in their errors for clusters of pixels. On the other hand, the pseudo-labels used by PRN during training reflect collective trends in bias in different patches.

We train PRN separately from the base network for several reasons. The first reason is that if a trained base network already exists (e.g., a state-of-the-art from prior work), then this reduces resource (e.g., electricity) consumption compared to retraining everything from scratch. The next reason is that once the base network is fixed, its logit maps for each training image won't change. Hence PRN can avoid expensive re-computation of the pseudo-labels it needs. Finally, the learning signal from the novel pseudo-labels used by PRN does not have a meaningful derivative with respect to the weights of the base network – the pseudo-labels are 0/1-valued numbers computed from the logit map of the base network; therefore the derivative with respect to the weights of the base network is either 0 or the delta function and hence does not work well with stochastic gradient descent-style optimization.

To summarize, our main contributions are:

- We propose PatchRefineNet (PRN), a post-processing network that sits on top of a base segmentation model and learns to correct its spatial biases.
- PRN uses a novel learning signal that is computed from binarizing each patch separately and optimally.
- PRN complements virtually any binary segmentation network. In our experiments across different base models, PRN consistently improves the mean Intersection over Union (mIoU) [53] and mean Boundary Accuracy (mBA) [9] by 2-3% over the base networks and hence there is good reason to believe that it can help future state-of-the-art networks improve their performance.
- We also explain how PRN can be extended to saliency detection, few-shot segmentation, and multi-class segmentation.

2. Related Work

Semantic Segmentation Architectures. Previous methods for semantic segmentation [28, 32, 41, 43, 45, 69, 72] have been successful in extracting contextual information with wide fields-of-view [4, 6, 17, 24, 44] along with FCN's [40] bottom-up approach for better segmentation quality. This includes feature pyramid methods [10, 19, 22, 39] that spa-

tially pool [39,68] feature maps of different receptive fields, or dilated convolutions [4, 7, 31, 62] with different dilation rates. Encoder-decoder models [2,7,31,36–38,46,49,50,54] have been widely used in semantic segmentation. The encoder reduces spatial resolution to capture high-level global semantics, followed by a decoder which restores spatial resolution. Skip connections [10, 42, 54] can be further added to recover lost spatial information in deeper layers. Self-attention [26, 47, 61, 74] has been used in segmentation networks to highlight salient features from context-rich skip connections and feature maps from deeper layers, where attention coefficients are more sensitive to local regions. Multi-scale context aggregation [3, 6, 21, 60] has proven to be efficient for integrating global and local features with two branches. Even though this alleviates higher memory usage arsing from using large output strides [4, 40], each branch has to be trained separately. PRN adopts the encoder-decoder architecture with skip connection and self-attention modules. Pyramid pooling is used for context aggregation at the bottleneck. PRN uses global and local branch decoders and allows for quick training and inference while being able to capture global and local structure from input logit maps.

Segmentation Refinement. FCN based methods typically do not generate very high-quality segmentation [10]. State-of-the-art network architectures have modules that increase field-of-view for constructing reliable context information [4, 6, 17, 18, 24, 26, 44, 63, 66, 68], and/or increase resolution of feature maps [5, 7, 55] to achieve better segmentation performance. Separate boundary refinement modules [51, 65] are also used to improve boundary accuracy. They are typically large models trained in an end-to-end fashion. However, they have limited refinement capability [7, 10, 64, 65] and inconsistencies [3, 64] still exist in their final binarized predictions due to inherent spatial biases [11, 30] in their output logit maps. Researchers have previously addressed the refinement process with postprocessing techniques like Conditional Random Fields (CRF) [3, 4, 34, 70] or region growing [14, 15]. Other methods use cascading [10, 33] and multi-scale context aggregation [8, 59] to generate high resolution segmentation maps. These methods aim at coarse-fine iterative refinement. However, to the best of our knowledge, no previous work has addressed refinement by focusing on correcting spatial biases [11, 30] from raw logit maps other than PRN. In comparison with PRN, above mentioned graphical methods [3, 4, 14, 15, 34, 70] cannot fix large errors and they adhere to local semantics without fully leveraging global structure. Cascading and context aggregation methods [10, 33] do not allow for single-stage training as individual patches are processed separately. Training time increases significantly with added levels of cascading. Additionally, as the patch size increases, memory usage also increases. On the contrary, PRN is a one-pass refinement module that allows for quick training and inference. Further, memory usage of PRN is constant for all patch sizes.

3. The Patch Refine Network (PRN)

We next discuss the PRN architecture (Sec. 3.1), how psuedo-labels are generation during training (Sec. 3.2), and the loss function used for training (Sec. 3.3). While the main focus here is on binary segmentation, we also discuss how to extend PRN to multi-class segmentation (Sec. 3.4).

3.1. Architecture

Given a trained binary segmentation model that serves as a base, PRN is designed to be a small, lightweight network that sits on top of this base and learns to correct its spatial biases. Its input is a logit map (produced by running an image through the base network) and its output is a logit map that can be thresholded at 0.5 to create binary predictions.

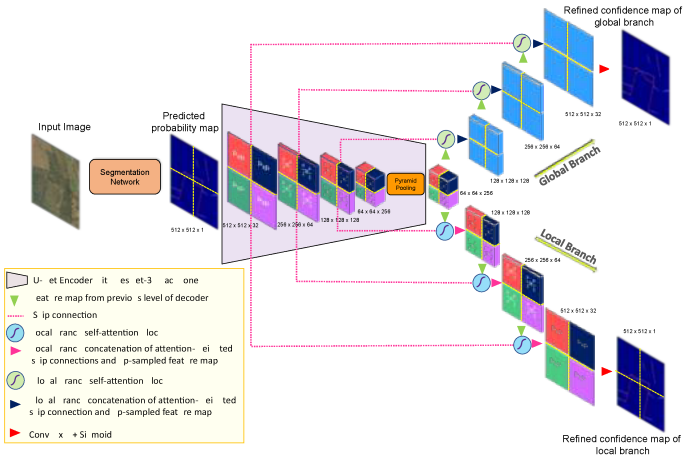

Figure 2. **PRN framework.** Example of PRN architecture configured for $k = 4$ patches. PRN has a U-Net [54] encoder with ResNet-34 [23] backbone with $\frac{1}{8}$ resolution scaling and pyramid pooling at the bottleneck. There are two decoder branches – a *global* and *local*.

It is common practice for such networks, which post-process the output of a base network [8, 59], to have both a *global* branch to capture the overall structure in an input and a *local* branch to analyze finer structures. PRN follows a similar strategy. While prior post-processing networks had to train local and global branches separately, PRN is able to jointly train the encoder and local/global decoders. The local and global branches both produce a logit map and the two maps are averaged during inference.

First, the logit map produced by the base segmentation network is resized to 512×512 and given as input to the PRN encoder. The output of the encoder, referred to as the *bottleneck*, is a feature map that has $\frac{1}{8}$ resolution scaling. This is simultaneously passed through the global and local

decoder branches. The global decoder takes the entire feature map at the bottleneck as input.

Before running the local branch, the output of the encoder is split into k disjoint patches. The value of k is determined by a patch-size parameter P as follows: $k = (512/P)^2$. Each patch is independently sent through the local branch to produce $1/k^{\text{th}}$ of the logit map. The full logit map of the local branch is then re-assembled from these pieces after the local branch processes all patches.

Encoder: We use a standard U-Net [54] encoder with ResNet-34 [23] backbone to extract features from the input logit map, as shown in Fig. 2. The spatial resolution decreases from 512 to 64 ($\frac{1}{8}^{th}$), while the number of features increases from 32 to 256 at the end of four encoding levels. After the fourth encoding level, pyramid pooling [68] with pooling sizes [1, 2, 4, 8] is used for rich global contextual features. The final resolution at the bottleneck, after pyramid pooling, is $64 \times 64 \times 256$.

Global Decoder Branch: The core purpose of this decoder is to capture global inter-patch semantics. This is a typical U-Net decoder with convolutional blocks, as shown in Fig. 2. We add a self-attention [47] layer at each decoding level before concatenating the skip connection from the previous encoder level with an upsampled feature map from the previous decoder layer. Self-attention filters highlight salient features from spatial-information-rich skip connections and context-rich decoder (deeper) layers. This branch is used to extract the relationship between patches.

Local Decoder Branch: The core purpose of this decoder is to capture local intra-patch semantics from each feature patch. A magnified version of the local branch self-attention block is shown in Fig. 3. The feature map from decoding level i has size $p \times p \times D$, where $p = \frac{P}{2^i}$, and D is the number of filters at decoding level i. It is broken down into k patches of size $\frac{p}{2} \times \frac{p}{2} \times D$. The skip connection from encoding level $i+1$ has size $2p \times 2p \times \frac{D}{2}$ and is also broken down into k patches of size $p \times p \times \frac{D}{2}$. Each patch from decoding level i is upsampled and concatenated with the attention-weighted patch from the encoding level $i+1$. Finally, processed patches are spatially merged to size $2p \times 2p \times \frac{D}{2}$, which is the output of decoding level $i+1$. This helps in capturing local semantics from each patch. The design of this branch allows the training to be performed simultaneously, as opposed to other patch aggregation and cascading methods [8, 59].

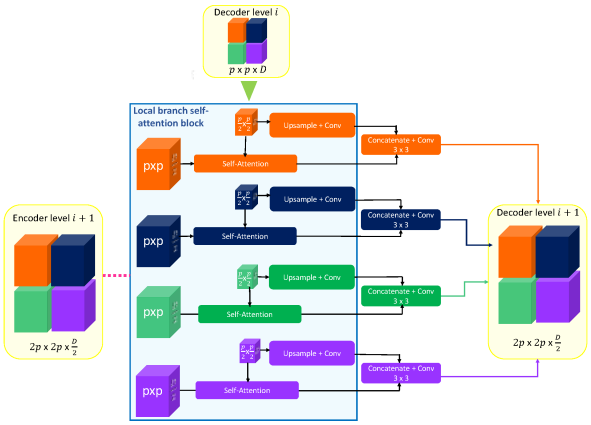

Figure 3. **Local branch self-attention block**, configured for $k = 4$ patches ($P = 256$). Inputs to the block are the feature map from decoding level i and skip connection from encoding level $i+1$.

Inference: During inference, the logit map output by the base network is passed as the input to the PRN. Both the local and global branches produce logit maps which are then averaged (resulting in the "final" logit map) and then thresholded at 0.5 (for final binarized predictions).

3.2. Generating Patch-Optimal Thresholded Maps

The data used for tuning the base network's hyperparameters also serves to train PRN. One learning signal (used in the loss function in Section 3.3) is the ground truth labeling Y of an image. The other is a novel set of "pseudo-labels". Let $\widehat{Y}$ be the logit map produced by the base network. This $\widehat{Y}$ is split into k patches. For each patch p_i, one finds the threshold that maximizes the mIoU for that patch (when the patch is binarized using the threshold). The resulting binarized patches are the pseudo-labels.

The intuition behind the pseudo-labels is that the most efficient way to minimize loss (e.g., binary cross-entry) between an output logit map and the pseudo-labels is to shift an entire patch from a logit map up or down. For example, in the case of the very last layer this is achieved by mainly focusing on the bias parameter of the layer. Thus pseudo-labels lets the network focus on aggregate properties (e.g., spatial biases) of the patch, whereas the ground truth signal makes the network focus on properties of individual pixels.

3.3. Loss Function

The loss function uses the two learning signals defined above. The global and local branches both use the same loss function, and the overall loss is the sum of the two. Hence we describe the loss $\mathbf{L}$ for one of the branches.

$\mathbf{L}$ is the weighted sum of two components, a loss $\mathbf{L}_{gt}$ with respect to the ground truth and a loss $\mathbf{L}_{ps}$ with respect to the pseudo-labels:

$$\mathbf{L} = \alpha \mathbf{L}_{ps} + (1-\alpha)\mathbf{L}_{gt} \quad \text{with } \alpha = 0.7, \tag{1}$$

where α was tuned based on 100 randomly augmented images from the DeepGlobe training set [12].

$\mathbf{L}_{gt}$ is the standard binary cross-entropy loss [67] between the ground truth and the logit map produced by the branch.

$\mathbf{L}_{ps}$ uses the pseudo-labels for the ground truth and can be written as a sum: $\mathbf{L}_{ps} = \mathbf{L}_{focal} + \mathbf{L}_{boundary}$, where $\mathbf{L}_{focal}$ is known as the *focal loss* [35] and $\mathbf{L}_{boundary}$ is known as the *boundary loss* [69]. Both focal and boundary loss are standard in image segmentation, however we use the pseudo-labels, generated for patch-size parameter P, in place of the ground truth in the computation of the losses. Focal loss [35] is a variation of binary cross-entropy loss that introduces a parameter γ (tuned using the same 100 DeepGlobe images as α in Equation 1). For a pixel i, let c_i be the output of the branch for that pixel (i.e., a logit value) and let $\widetilde{y}_i$ be the pseudo-label. Then

$$\mathbf{L}_{focal} = -\sum_{i\,:\,\widetilde{y}_i=1} (1-c_i)^{\gamma}\log(c_i) - \sum_{i\,:\,\widetilde{y}_i=0} c_i^{\gamma}\log(1-c_i)$$

Boundary loss [69] is designed to improve predictions at boundary pixels. It is computed as follows. Let C be the matrix corresponding to the logit map output by a branch. The squashed Laplace operator [69] applied to C is:

$$\mathrm{abs}(\tanh(\mathrm{conv}(C,K))) \quad \text{where } k = \begin{pmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{pmatrix}$$

The boundary loss $\mathbf{L}_{boundary}$ [69] is the defined as the binary cross entropy between the squashed Laplace operator applied to C and the squashed Laplace operator applied to the target labels, which in our case are the pseudo-labels.

3.4. Extension to Multi-Class Segmentation

In this section, we explain how this technique could be extended to multiclass semantic segmentation with m classes. The output at each pixel, instead of being a single logit, is now a m-dimensional vector produced by the softmax activation. If we let $\vec{x}_i$ denote the pre-activation at pixel i, then the output at the pixel is $\mathrm{softmax}(\vec{x}_i)$.

The pseudo-label for a pixel becomes a m-dimensional one-hot encoding vector. During training it can be generated as follows. Previously, the best threshold was used to binarize each image patch. In the multiclass setting, the threshold is replaced by a m-dimensional vector $\vec{t}$ and the "pseudo-label class" for a pixel is chosen by the formula: $\arg\max_j(\vec{t}[j] + \mathrm{softmax}(\vec{x}_i)[j])$ – this is the same as translating the softmax by the vector $\vec{t}$ and choosing class j if the j^{th} component is the largest. The pseudo-label is the one-hot encoding of the chosen class.

The difficulty here is in choosing the optimal $\vec{t}$ for each image patch in the training data. In the binary case, we were only dealing with a threshold, and it was easy to try different numbers between 0 and 1. However, this becomes inefficient when searching for the optimal vector $\vec{t}$. Performing this search efficiently is part of our future work, and our goal in this paper is to evaluate how well PRN works in the binary segmentation setting.

4. Experiments

In this section, we evaluate the ability of PRN to improve the prediction of a base binary segmentation network. We consider a variety of datasets and base networks (including current and former state-of-the-art segmentation models) along with other postprocessing methods. Overall, PRN consistently improves performance in mIoU by approximately 2-3% and thus is likely to help future models improve their predictions as well, by reducing their spatial biases.

4.1. Datasets

We use the following four datasets for evaluation: DeepGlobe [12] [1], Kvasir-SEG [27], DUTS [57], and FSS-1000 [32] on three types of tasks: binary segmentation (DeepGlobe, Kvasir-SEG), saliency detection (DUTS), and few-shot segmentation (FSS-1000).

DeepGlobe [12] is a large-scale road extraction dataset that contains 6226 labeled images. We divide this into 4980 training images, 996 validation images, and 250 test images. Kvasir-SEG [27] is a large-scale polyp segmentation dataset with 1000 labeled images. DUTS [57] contains 10553 images for training and 5019 images for evaluation. We divide these 5019 images into 4015 validation images and 1004 test images. FSS-1000 [32] contains 1000 classes with 10 images each. We divide the 1000 classes into 760 classes for training, 192 classes for validation, and 48 classes for testing. Each class contains 10 images out of which we use 5 images as support (labeled images to generalize from for few-shot learning) and the other 5 as query (test images).

4.2. Evaluation Criteria

Similar to prior work in binary segmentation, we use mean Intersection over Union [53] (mIoU) and mean Boundary Accuracy (mBA) [9] as the evaluation metrics. mBA, also called boundary mIoU, is a new measure proposed by [9] which has a weaker bias toward large objects than mIoU. It neither over-penalizes nor ignores errors in small objects. Given the matrix of ground truth pixel labels and (binarized) predicted labels, Boundary mIoU first computes the set of the pixels that are within a distance d from each contour (computed from [48]) in the ground truth and in the predictions and then computes mIoU of these two sets. We use $d = 15$ as recommended in [9]. We evaluate the performance of PRN on Saliency detection [69] using mean absolute error (MAE), along with mIoU and mBA.

4.3. Implementation Details

The base networks are trained according to the code and implementation details provided in the respective papers.

[1] It must be noted that this is DeepGlobe Road Extraction dataset, not DeepGlobe land cover classification dataset.

The datasets we use are divided into training, validation, and test sets as discussed in Sec. 4.1. The train set is used to train the base model. The validation set is used to tune the hyperparameters of the base model and to train PRN (the validation set is never used for reporting). To make sure comparisons are fair, we also try settings where the base model include the validation data in training (Sec. 4.4.1). The patch size used by the local branch of PRN is controlled by the parameter P. The best choice is $P = 64$, which is determined by a hyperparameter search on 100 randomly augmented training images (see supplementary material for additional details). This results in the local branch dividing the input logit map into sixty-four patches of size 64×64. Since the testing set was not used at all for choosing patch size, it is appropriate to use $P = 64$ in the rest of our experiments. Finally, the test set is used for reporting results.

Data augmentations such as random rotation, and horizontal and vertical flips are used for training the models. PRN is trained with the Adam [29] optimizer with an initial learning rate of $8e^{-4}$, batch size of 4, and for a maximum of 300 epochs on an NVIDIA 2080 Ti GPU. The learning rate is decreased until $5e^{-8}$. We use early stopping if its training loss does not decrease for 10 epochs.

4.4. Ablation Experiments

We first present ablation studies using the DeepGlobe [12] dataset and base network DLinkNet [72].

4.4.1 Role of the validation set.

Ordinarily, the base model would train on the training set and tune hyperparameters on the validation set, which is also used to train PRN (we emphasise that *results* are reported on the *test* set only, which is disjoint from validation and train). PRN uses the validation data because this is where the spatial bias of the base models become apparent.

This raises the question of whether it is a fair setup – would it be better to simply add the validation data to the base model's training set and not use PRN? To answer this question, we consider the following 3 cases. (**A**) The base network trains on training data and tunes hyperparameters on validation data; PRN is not used. (**B**): The base network is trained using the combined training and validation data; we use the default hyperparameters from the DLinkNet github repository [16]; PRN is not used. (**C**): The base network trains on training data and tunes hyperparameters on validation data; PRN is then trained on the validation data. The results, reported on the test set (disjoint from train and validation) are shown in Table 1.

As we can see, reserving some data for hyper-parameter tuning is beneficial to the base network (case A improves upon case B). Re-using this validation set to train PRN shows a further, significant boost (case C is by far the best).

Experiments	**DeepGlobe [12] test-set** mIoU (%)	mBA (%)
A: Train on train set, tune on validation, no PRN	61.3	49.8
B: Train on train and validation set, no PRN	59.7	48.4
C: Train on train set, tune on validation, yes PRN	**64.4**	**56.6**

Table 1. Evaluating the role of the validation set.

This validates our proposed setup for how different parts of the data are used.

4.4.2 Ablation study of PRN design.

We next consider an ablation study of the rest of the design of PRN, including the benefit of using of global/local branches and a loss function based on pseudo-labels.

It is becoming increasingly common to use global and local branches to improve segmentation quality [8, 59]. In the case of PRN, where we want to detect and correct patch-specific spatial biases, local branches are clearly necessary from the design perspective. At the top of Table 2, we compare performance when PRN includes a global branch only, local branch only, and both branches together. As expected, the local branch is much more important than the global branch, with roughly a 2% better mIoU and 4% better mean boundary accuracy. Also, as expected, there is a very slight performance boost when the global branch is added to the local branch, as this allows PRN to incorporate wider context information from the global branch.

Configuration	**DeepGlobe [72] test-set** mIoU (%)	mBA (%)
Base Network: D-LinkNet [12]	61.3	49.8
Ablation of network design		
Global branch only	$61.7_{\uparrow 0.4}$	$52.3_{\uparrow 2.5}$
Local branch only	$63.5_{\uparrow 2.2}$	$56.1_{\uparrow 6.3}$
Local + Global (ours)	$\mathbf{64.4_{\uparrow 3.1}}$	$\mathbf{56.6_{\uparrow 6.8}}$
Ablation of total loss function		
$\mathbf{L}_{gt}$ only	$62.0_{\uparrow 0.7}$	$52.8_{\uparrow 3.0}$
$\mathbf{L}_{ps}$ only	$63.8_{\uparrow 2.5}$	$57.2_{\uparrow 7.4}$
$\mathbf{L}_{gt}$ + $\mathbf{L}_{ps}$ (ours)	$\mathbf{64.4_{\uparrow 3.1}}$	$\mathbf{56.6_{\uparrow 6.8}}$
Ablation of Region-specific loss $\mathbf{L_{ps}}$		
$\mathbf{L}_{focal}$ only	$64.0_{\uparrow 2.7}$	$51.9_{\uparrow 2.1}$
$\mathbf{L}_{boundary}$ only	$62.2_{\uparrow 0.9}$	$55.4_{\uparrow 5.6}$
$\mathbf{L}_{focal} + \mathbf{L}_{boundary}$ (ours)	$\mathbf{64.4_{\uparrow 3.1}}$	$\mathbf{56.6_{\uparrow 6.8}}$

Table 2. Ablation results for the design of PRN ($P = 64$).

Now, recall that the loss function in each branch is a sum of two losses $\mathbf{L}_{gt}$ whose learning signal comes from the ground truth and $\mathbf{L}_{ps}$ which comes from our pseudo-labels.

The middle section of Table 2 shows the results of using only the ground truth ($\mathbf{L}_{gt}$), only the pseudo-labels ($\mathbf{L}_{ps}$), or both ($\mathbf{L}_{gt} + \mathbf{L}_{ps}$). Again we see that the pseudo-labels are more important than using the ground truth, probably because the base network is already trained with the ground truth signal, while the pseudo-labels summarize new information about systematic biases (as explained in Sec. 3.2). As expected, combining the two losses leads to a slight improvement over using pseudo-labels alone since the ground truth does contain information not present in pseudo-labels.

Finally, the loss over pseudo-labels, which is designed to correct patch-wise spatial biases is a mixture of focal loss [35] and boundary loss [69]. Both are used in the literature to improve segmentation on fine structures, with boundary loss focusing on the boundary. As we can tell from the bottom of Table 2, focal loss is better at improving mIoU while boundary loss is better at improving mean boundary accuracy, which is consistent with prior work. The combination of the two losses gives us the best of both worlds.

4.5. Performance Evaluation

4.5.1 Binary segmentation.

We next evaluate the improvement that PRN provides when combined with a variety of state-of-the-art and former state-of-the-art networks for binary segmentation on the DeepGlobe [2] and Kvasir-SEG datasets. Table 3 shows that PRN provides consistent improvement by at least 2.3% in mIoU and 2.6% mBA on both datasets for all networks, illustrating that they all have spatial bias, which PRN addresses. This supports the hypothesis that PRN is likely to help future networks to further improve their performance. Fig. 4 shows qualitative examples. The first two rows come from the DeepGlobe test set with CoANet [41] as the base network; the task is to identify roads in the image. The last two rows are from the Kvasir-SEG test data with SSFormer-S [56] as the base; the task is to identify polyps. The first two columns show the logit map and binarized prediction, respectively, of the base network. The yellow boxes highlight areas of false positives and false negatives. The next two columns show the logit map and binarized prediction after PRN de-biases the base networks. The last column shows the ground truth. For example, in the first row, the left-most yellow box identifies a region where the base network missed part of a road, resulting in two disconnected road segments; this is a negative bias in that region that PRN fixes. In the second row, the base network predicts that the roads have an 'A' shape but the cross-bar is a false positive that gets removed by PRN. The corrections made by PRN are more clearly visible in the last two rows.

[2]It must be noted that this is DeepGlobe Road Extraction dataset, not DeepGlobe land cover classification dataset. So, the results reported cannot be compared with papers using the latter.

DeepGlobe [12] test-set		
Methods	mIoU (%)	mBA (%)
U-Net [54]	55.8	37.6
(+) PRN	**60.9**$_{\uparrow 5.1}$	**47.4**$_{\uparrow 9.8}$
DeepLabV3+ [4]	59.2	47.6
(+) PRN	**61.9**$_{\uparrow 2.7}$	**55.9**$_{\uparrow 8.3}$
PSPNet [68]	59.8	48.2
(+) PRN	**62.4**$_{\uparrow 2.6}$	**56.6**$_{\uparrow 8.4}$
D-LinkNet [72]	61.3	49.8
(+) PRN	**64.4**$_{\uparrow 3.1}$	**56.6**$_{\uparrow 6.8}$
GLNet [8]	62.8	52.6
(+) PRN	**65.4**$_{\uparrow 2.6}$	**57.9**$_{\uparrow 5.3}$
ISDNet [20]	64.8	54.8
(+) PRN	**67.3**$_{\uparrow 2.5}$	**59.2**$_{\uparrow 4.4}$
CoANet [41]	67.9	58.4
(+) PRN	**70.6**$_{\uparrow 2.7}$	**62.1**$_{\uparrow 3.7}$
Kvasir-SEG [27] test-set		
U-Net [54]	41.5	38.8
(+) PRN	**47.8**$_{\uparrow 6.3}$	**46.3**$_{\uparrow 7.5}$
ResUnet [13]	46.8	45.7
(+) PRN	**52.9**$_{\uparrow 6.1}$	**52.5**$_{\uparrow 6.8}$
ResUnet++ [28]	55.9	56.8
(+) PRN	**61.7**$_{\uparrow 5.8}$	**62.9**$_{\uparrow 6.1}$
SSFormer-S [56]	86.8	69.7
(+) PRN	**89.1**$_{\uparrow 2.3}$	**72.3**$_{\uparrow 2.6}$

Table 3. How PRN helps base networks for binary segmentation.

4.5.2 Comparison with other post-processing methods

Although the literature on post-processing methods is very sparse, DenseCRF [70] and CascadePSP [10] are two notable postprocessing techniques for improving binary segmentation. Our first comparison, to DenseCRF, shows that PRN is much better at improving both mIoU and mBA. Due to space restrictions, a small subset of our results in shown in Table 4. More extensive comparisons with DenseCRF for all datasets can be found in the supplementary material.

DeepGlobe [12] test-set		
Methods	mIoU (%)	mBA (%)
CoANet [41]	67.9	58.4
(+) DenseCRF [70]	69.0$_{\uparrow 1.1}$	59.6$_{\uparrow 1.2}$
(+) PRN	**70.6**$_{\uparrow 2.7}$	**62.1**$_{\uparrow 3.7}$

Table 4. Comparison to DenseCRF [70] postprocessing.

.

CascadePSP [10] is another post-processing technique that supports several different configurations, such as number of cascade levels in the global step and different image crop sizes. In Table 5, we compare PRN with CascadePSP with different configurations. As an ablation ex-

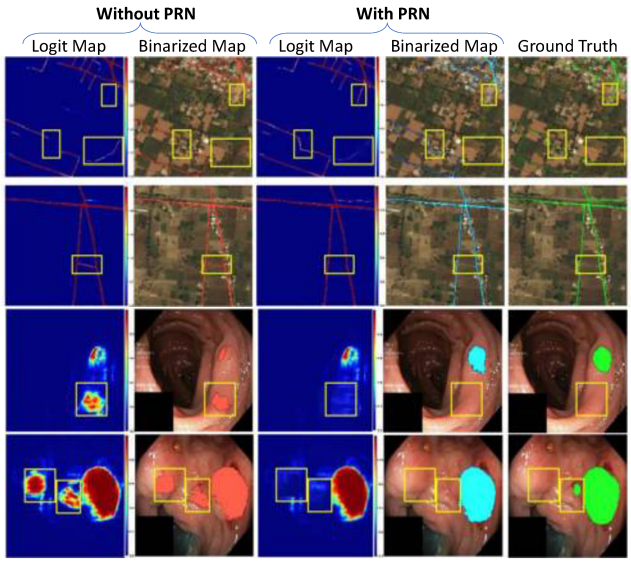

Figure 4. Qualitative examples of improvement due to PRN. Rows 1 & 2: images from DeepGlobe with CoANet as the base network. Rows 3 & 4: images from Kvasir with SSFormer-S as the base network. Ground truth: last column. Yellow boxes represent areas where PRN causes most improvement. Column 1: logit map of the base network. Column 2: binarized predictions of base network. Column 3: logit map output by PRN. Column 4: binarized predictions from PRN.

DeepGlobe [12] test-set		
Base Network CoANET mIoU (%): 67.9 %		
Configuration	mIoU(%)	memory usage (GB)
Levels of cascading for Global step		
(+) CascadePSP (1-level)	$68.1_{\uparrow 0.2}$	1.03
(+) CascadePSP (3-level)	$68.8_{\uparrow 0.9}$	1.03
(+) PRN	$\mathbf{70.6}_{\uparrow \mathbf{2.7}}$	**1.03**
Addition of Local step for different image crop sizes L		
(+) CascadePSP (L=512)	$68.9_{\uparrow 1.0}$	2.12
(+) CascadePSP (L=900)	$69.2_{\uparrow 1.3}$	3.46
(+) CascadePSP (L=1024)	$69.7_{\uparrow 1.8}$	4.08
(+) PRN	$\mathbf{70.6}_{\uparrow \mathbf{2.7}}$	**1.03**

Table 5. **Quantitative results comparing PRN ($P = 64$) with CascadePSP [10] on DeepGlobe test dataset**.

periment, we first consider just the global step of CascadePSP and change the number of cascade levels. This provides very marginal improvement over the base network and it is clearly outperformed by PRN (top half of Table 5). Then we add the local step for CascadePSP and vary the image crop size parameter that it uses. This continues to improve the performance of CascadePSP, but it is still dominated by PRN (bottom half of Table 5). The memory usage of CascadePSP grows with crop size and even when it needs 4 times as much memory as PRN, it is still outperformed by PRN.

4.5.3 Saliency Detection on DUTS

We next consider saliency detection (identifing the pixels of the salient objects in an image) using the DUTS dataset [57] and one of its state-of-the-art methods, PFAN [69], as the base network. The results are shown in Table 6. Adding PRN resulted in significant improvement of +3.8% and +7.4% in mIoU and mBA, again showing the potential of PRN in improving different kinds of networks.

DUTS [57] test-set			
Methods	mIoU (%)	mBA (%)	MAE
RFCN [58]	52.8	40.7	0.0897
(+) PRN	$\mathbf{57.1}_{\uparrow \mathbf{4.3}}$	$\mathbf{48.5}_{\uparrow \mathbf{7.8}}$	**0.0807**
PFAN [69]	66.1	51.2	0.0452
(+) PRN	$\mathbf{69.9}_{\uparrow \mathbf{3.8}}$	$\mathbf{58.6}_{\uparrow \mathbf{7.4}}$	**0.0386**

Table 6. **PRN and DUTS Saliency detection test dataset**.

4.5.4 Few-shot segmentation on FSS-1000 [32].

Finally, in Table 7, we apply PRN to few-shot segmentation over the FSS-1000 [32] dataset with EfficientLab [25] and ARN [32] as the base networks. Again there is consistent improvement of at least +1.3% and +2.1% in mIoU and mBA.

FSS-1000 [32] test-set		
Methods	mIoU (%)	mBA (%)
Adapted Relation Network [32]	80.1	69.8
(+) PRN	$\mathbf{82.7}_{\uparrow \mathbf{2.6\%}}$	$\mathbf{72.9}_{\uparrow \mathbf{3.1\%}}$
EfficientLab [25]	82.8	71.1
(+) PRN	$\mathbf{84.1}_{\uparrow \mathbf{1.3\%}}$	$\mathbf{73.2}_{\uparrow \mathbf{2.1\%}}$

Table 7. **PRN and FSS-1000 Saliency detection test dataset**.

5. Conclusion

We proposed PatchRefineNet (PRN), a post-processing network that sits on top of a base segmentation model and learns to correct its spatial biases. PRN uses a novel learning signal that is computed from binarizing each patch separately. PRN complements virtually any binary segmentation network and also works with saliency detection. In our experiments across different base models, PRN consistently helps the base networks improve both mIoU and mBA by over 2-3 %. [3] This work was supported by the Google AI Impact Challenge under Grant 1904-57775, NSF awards CNS-1702760, and CNS-1931686.

[3] Additional experiments can be found in the supplemenatry material.

References

[1] Marjan Alirezaie, Martin Längkvist, Michael Sioutis, and Amy Loutfi. Semantic referee: A neural-symbolic framework for enhancing geospatial semantic segmentation. *Semantic Web*, 10(5):863–880, 2019. 1

[2] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 39(12):2481–2495, 2017. 1, 3

[3] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Semantic image segmentation with deep convolutional nets and fully connected crfs. *arXiv preprint arXiv:1412.7062*, 2014. 1, 3

[4] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis and machine intelligence*, 40(4):834–848, 2017. 2, 3, 7

[5] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation. *arXiv preprint arXiv:1706.05587*, 2017. 3

[6] Liang-Chieh Chen, Yi Yang, Jiang Wang, Wei Xu, and Alan L Yuille. Attention to scale: Scale-aware semantic image segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3640–3649, 2016. 2, 3

[7] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *Proceedings of the European conference on computer vision (ECCV)*, pages 801–818, 2018. 3

[8] Wuyang Chen, Ziyu Jiang, Zhangyang Wang, Kexin Cui, and Xiaoning Qian. Collaborative global-local networks for memory-efficient segmentation of ultra-high resolution images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8924–8933, 2019. 3, 4, 6, 7

[9] Bowen Cheng, Ross Girshick, Piotr Dollár, Alexander C Berg, and Alexander Kirillov. Boundary iou: Improving object-centric image segmentation evaluation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15334–15342, 2021. 2, 5

[10] Ho Kei Cheng, Jihoon Chung, Yu-Wing Tai, and Chi-Keung Tang. Cascadepsp: Toward class-agnostic and very high-resolution segmentation via global and local refinement. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8890–8899, 2020. 2, 3, 7, 8

[11] Steffen Czolbe, Kasra Arnavaz, Oswin Krause, and Aasa Feragen. Is segmentation uncertainty useful? In *Information Processing in Medical Imaging: 27th International Conference, IPMI 2021, Virtual Event, June 28–June 30, 2021, Proceedings 27*, pages 715–726. Springer, 2021. 3

[12] Ilke Demir, Krzysztof Koperski, David Lindenbaum, Guan Pang, Jing Huang, Saikat Basu, Forest Hughes, Devis Tuia, and Ramesh Raskar. Deepglobe 2018: A challenge to parse the earth through satellite images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 172–181, 2018. 1, 4, 5, 6, 7, 8

[13] Foivos I Diakogiannis, François Waldner, Peter Caccetta, and Chen Wu. Resunet-a: A deep learning framework for semantic segmentation of remotely sensed data. *ISPRS Journal of Photogrammetry and Remote Sensing*, 162:94–114, 2020. 7

[14] Philipe Ambrozio Dias and Henry Medeiros. Semantic segmentation refinement by monte carlo region growing of high confidence detections. In *Asian Conference on Computer Vision*, pages 131–146. Springer, 2018. 3

[15] Philipe A Dias and Henry Medeiros. Probabilistic semantic segmentation refinement by monte carlo region growing. *arXiv preprint arXiv:2005.05856*, 2020. 3

[16] DLinkNet-GitHub. GitHub - zlckanata/DeepGlobe-Road-Extraction-Challenge: D-LinkNet: LinkNet with Pretrained Encoder and Dilated Convolution for High Resolution Satellite Imagery Road Extraction — github.com. https://github.com/zlckanata/DeepGlobe-Road-Extraction-Challenge. [Accessed 24-08-2023]. 6

[17] Clement Farabet, Camille Couprie, Laurent Najman, and Yann LeCun. Learning hierarchical features for scene labeling. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1915–1929, 2012. 2, 3

[18] Jun Fu, Jing Liu, Haijie Tian, Yong Li, Yongjun Bao, Zhiwei Fang, and Hanqing Lu. Dual attention network for scene segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3146–3154, 2019. 3

[19] Christopher Funk, Savinay Nagendra, Jesse Scott, Bharadwaj Ravichandran, John H Challis, Robert T Collins, and Yanxi Liu. Learning dynamics from kinematics: Estimating 2d foot pressure maps from video frames. *arXiv preprint arXiv:1811.12607*, 2018. 2

[20] Shaohua Guo, Liang Liu, Zhenye Gan, Yabiao Wang, Wuhao Zhang, Chengjie Wang, Guannan Jiang, Wei Zhang, Ran Yi, Lizhuang Ma, et al. Isdnet: Integrating shallow and deep networks for efficient ultra-high resolution segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4361–4370, 2022. 7

[21] Bharath Hariharan, Pablo Arbeláez, Ross Girshick, and Jitendra Malik. Hypercolumns for object segmentation and fine-grained localization. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 447–456, 2015. 3

[22] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. *IEEE transactions on pattern analysis and machine intelligence*, 37(9):1904–1916, 2015. 2

[23] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 3, 4

[24] Xuming He, Richard S Zemel, and Miguel A Carreira-Perpinán. Multiscale conditional random fields for image labeling. In *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.*, volume 2, pages II–II. IEEE, 2004. 2, 3

[25] Sean M Hendryx, Andrew B Leach, Paul D Hein, and Clayton T Morrison. Meta-learning initializations for image segmentation. *arXiv preprint arXiv:1912.06290*, 2019. 8

[26] Lang Huang, Yuhui Yuan, Jianyuan Guo, Chao Zhang, Xilin Chen, and Jingdong Wang. Interlaced sparse self-attention for semantic segmentation. *arXiv preprint arXiv:1907.12273*, 2019. 3

[27] Debesh Jha, Pia H Smedsrud, Michael A Riegler, Pål Halvorsen, Thomas de Lange, Dag Johansen, and Håvard D Johansen. Kvasir-seg: A segmented polyp dataset. In *International Conference on Multimedia Modeling*, pages 451–462. Springer, 2020. 1, 5, 7

[28] Debesh Jha, Pia H Smedsrud, Michael A Riegler, Dag Johansen, Thomas De Lange, Pål Halvorsen, and Håvard D Johansen. Resunet++: An advanced architecture for medical image segmentation. In *2019 IEEE International Symposium on Multimedia (ISM)*, pages 225–2255. IEEE, 2019. 2, 7

[29] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 6

[30] Michael C Krygier, Tyler LaBonte, Carianne Martinez, Chance Norris, Krish Sharma, Lincoln N Collins, Partha P Mukherjee, and Scott A Roberts. Quantifying the unknown impact of segmentation uncertainty on image-based simulations. *Nature communications*, 12(1):5414, 2021. 3

[31] Hanchao Li, Pengfei Xiong, Jie An, and Lingxue Wang. Pyramid attention network for semantic segmentation. *arXiv preprint arXiv:1805.10180*, 2018. 3

[32] Xiang Li, Tianhan Wei, Yau Pun Chen, Yu-Wing Tai, and Chi-Keung Tang. Fss-1000: A 1000-class dataset for few-shot segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2869–2878, 2020. 1, 2, 5, 8

[33] Guosheng Lin, Anton Milan, Chunhua Shen, and Ian Reid. Refinenet: Multi-path refinement networks for high-resolution semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1925–1934, 2017. 3

[34] Guosheng Lin, Chunhua Shen, Anton Van Den Hengel, and Ian Reid. Efficient piecewise training of deep structured models for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3194–3203, 2016. 3

[35] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017. 5, 7

[36] Chenxi Liu, Liang-Chieh Chen, Florian Schroff, Hartwig Adam, Wei Hua, Alan L Yuille, and Li Fei-Fei. Auto-deeplab: Hierarchical neural architecture search for semantic image segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 82–92, 2019. 3

[37] Jiangtao Liu, Chaopeng Shen, Te Pei, Kathryn Lawson, Daniel Kifer, Savinay Nagendra, and Srikanth Banagere Manjunatha. A new rainfall-induced deep learning strategy for landslide susceptibility prediction. In *AGU Fall Meeting Abstracts*, volume 2021, pages NH35E–0504, 2021. 3

[38] Shu Liu, Lu Qi, Haifang Qin, Jianping Shi, and Jiaya Jia. Path aggregation network for instance segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8759–8768, 2018. 3

[39] Wei Liu, Andrew Rabinovich, and Alexander C Berg. Parsenet: Looking wider to see better. *arXiv preprint arXiv:1506.04579*, 2015. 1, 2, 3

[40] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015. 2, 3

[41] Jie Mei, Rou-Jing Li, Wang Gao, and Ming-Ming Cheng. Coanet: Connectivity attention network for road extraction from satellite imagery. *IEEE Transactions on Image Processing*, 30:8540–8552, 2021. 1, 2, 7

[42] Savinay Nagendra, S Banagere Manjunatha, Chaopeng Shen, Daniel Kifer, and Te Pei. An efficient deep learning mechanism for cross-region generalization of landslide events. In *AGU Fall Meeting Abstracts*, volume 2020, pages NH030–0010, 2020. 3

[43] Savinay Nagendra, Daniel Kifer, Benjamin Mirus, Te Pei, Kathryn Lawson, Srikanth Banagere Manjunatha, Weixin Li, Hien Nguyen, Tong Qiu, Sarah Tran, et al. Constructing a large-scale landslide database across heterogeneous environments using task-specific model updates. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 15:4349–4370, 2022. 2

[44] Savinay Nagendra, Nikhil Podila, Rashmi Ugarakhod, and Koshy George. Comparison of reinforcement learning algorithms applied to the cart-pole problem. In *2017 international conference on advances in computing, communications and informatics (ICACCI)*, pages 26–32. IEEE, 2017. 2, 3

[45] Savinay Nagendra, Chaopeng Shen, and Daniel Kifer. Threshnet: Segmentation refinement inspired by region-specific thresholding. *arXiv preprint arXiv:2211.06560*, 2022. 2

[46] Hyeonwoo Noh, Seunghoon Hong, and Bohyung Han. Learning deconvolution network for semantic segmentation. In *Proceedings of the IEEE international conference on computer vision*, pages 1520–1528, 2015. 3

[47] Ozan Oktay, Jo Schlemper, Loic Le Folgoc, Matthew Lee, Mattias Heinrich, Kazunari Misawa, Kensaku Mori, Steven McDonagh, Nils Y Hammerla, Bernhard Kainz, et al. Attention u-net: Learning where to look for the pancreas. *arXiv preprint arXiv:1804.03999*, 2018. 3, 4

[48] OpenSource. OpenCV: Contours : Getting Started — docs.opencv.org. https://docs.opencv.org/3.4/d4/d73/tutorial_py_contours_begin.html. [Accessed 23-08-2023]. 5

[49] Te Pei, Savinay Nagendra, Srikanth Banagere Manjunatha, Guanlin He, Daniel Kifer, Tong Qiu, and Chaopeng Shen.

Utilizing an interactive ai-empowered web portal for landslide labeling for establishing a landslide database in washington state, usa. In *EGU General Assembly Conference Abstracts*, pages EGU21–13974, 2021. 3

[50] Te Pei, Savinay Nagendra, Srikanth Banagere Manjunatha, Guanlin He, Tong Qiu, Daniel Kifer, and Chaopeng Shen. Cloud-based interactive database management suite integrated with deep learning-based annotation tool for landslide mapping. In *AGU Fall Meeting 2020*. AGU, 2020. 3

[51] Chao Peng, Xiangyu Zhang, Gang Yu, Guiming Luo, and Jian Sun. Large kernel matters–improve semantic segmentation by global convolutional network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4353–4361, 2017. 3

[52] Dzung L Pham, Chenyang Xu, and Jerry L Prince. A survey of current methods in medical image segmentation. *Annual review of biomedical engineering*, 2(3):315–337, 2000. 1

[53] Hamid Rezatofighi, Nathan Tsoi, JunYoung Gwak, Amir Sadeghian, Ian Reid, and Silvio Savarese. Generalized intersection over union: A metric and a loss for bounding box regression. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 658–666, 2019. 2, 5

[54] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015. 1, 3, 4, 7

[55] Ke Sun, Yang Zhao, Borui Jiang, Tianheng Cheng, Bin Xiao, Dong Liu, Yadong Mu, Xinggang Wang, Wenyu Liu, and Jingdong Wang. High-resolution representations for labeling pixels and regions. *arXiv preprint arXiv:1904.04514*, 2019. 3

[56] Jinfeng Wang, Qiming Huang, Feilong Tang, Jia Meng, Jionglong Su, and Sifan Song. Stepwise feature fusion: Local guides global. In *Medical Image Computing and Computer Assisted Intervention–MICCAI 2022: 25th International Conference, Singapore, September 18–22, 2022, Proceedings, Part III*, pages 110–120. Springer, 2022. 7

[57] Lijun Wang, Huchuan Lu, Yifan Wang, Mengyang Feng, Dong Wang, Baocai Yin, and Xiang Ruan. Learning to detect salient objects with image-level supervision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 136–145, 2017. 1, 5, 8

[58] Linzhao Wang, Lijun Wang, Huchuan Lu, Pingping Zhang, and Xiang Ruan. Saliency detection with recurrent fully convolutional networks. In *European conference on computer vision*, pages 825–841. Springer, 2016. 8

[59] Tong Wu, Zhenzhen Lei, Bingqian Lin, Cuihua Li, Yanyun Qu, and Yuan Xie. Patch proposal network for fast semantic segmentation of high-resolution images. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 12402–12409, 2020. 3, 4, 6

[60] Fangting Xia, Peng Wang, Liang-Chieh Chen, and Alan L Yuille. Zoom better to see clearer: Human and object parsing with hierarchical auto-zoom net. In *European Conference on Computer Vision*, pages 648–663. Springer, 2016. 3

[61] Linwei Ye, Mrigank Rochan, Zhi Liu, and Yang Wang. Cross-modal self-attention network for referring image segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10502–10511, 2019. 3

[62] Fisher Yu and Vladlen Koltun. Multi-scale context aggregation by dilated convolutions. *arXiv preprint arXiv:1511.07122*, 2015. 3

[63] Yuhui Yuan, Lang Huang, Jianyuan Guo, Chao Zhang, Xilin Chen, and Jingdong Wang. Ocnet: Object context network for scene parsing. *arXiv preprint arXiv:1809.00916*, 2018. 3

[64] Yuhui Yuan, Jingyi Xie, Xilin Chen, and Jingdong Wang. Segfix: Model-agnostic boundary refinement for segmentation. In *European Conference on Computer Vision*, pages 489–506. Springer, 2020. 3

[65] Chi Zhang, Guosheng Lin, Fayao Liu, Rui Yao, and Chunhua Shen. Canet: Class-agnostic segmentation networks with iterative refinement and attentive few-shot learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5217–5226, 2019. 3

[66] Hang Zhang, Han Zhang, Chenguang Wang, and Junyuan Xie. Co-occurrent features in semantic segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 548–557, 2019. 3

[67] Zhilu Zhang and Mert Sabuncu. Generalized cross entropy loss for training deep neural networks with noisy labels. *Advances in neural information processing systems*, 31, 2018. 4

[68] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2881–2890, 2017. 3, 4, 7

[69] Ting Zhao and Xiangqian Wu. Pyramid feature attention network for saliency detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3085–3094, 2019. 2, 5, 7, 8

[70] Shuai Zheng, Sadeep Jayasumana, Bernardino Romera-Paredes, Vibhav Vineet, Zhizhong Su, Dalong Du, Chang Huang, and Philip HS Torr. Conditional random fields as recurrent neural networks. In *Proceedings of the IEEE international conference on computer vision*, pages 1529–1537, 2015. 3, 7

[71] Zhuo Zheng, Yanfei Zhong, Junjue Wang, and Ailong Ma. Foreground-aware relation network for geospatial object segmentation in high spatial resolution remote sensing imagery. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4096–4105, 2020. 1

[72] Lichen Zhou, Chuang Zhang, and Ming Wu. D-linknet: Linknet with pretrained encoder and dilated convolution for high resolution satellite imagery road extraction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 182–186, 2018. 1, 2, 6, 7

[73] Zongwei Zhou, Md Mahfuzur Rahman Siddiquee, Nima Tajbakhsh, and Jianming Liang. Unet++: A nested u-net architecture for medical image segmentation. In *Deep learning in medical image analysis and multimodal learning for clinical decision support*, pages 3–11. Springer, 2018. 1

[74] Zhen Zhou, Yan Zhou, Dongli Wang, Jinzhen Mu, and Haibin Zhou. Self-attention feature fusion network for semantic segmentation. *Neurocomputing*, 453:50–59, 2021. 3