Smart Structures and Systems, *Vol. 31*, *No. 4* (*2023*) *365-381*
https://doi.org/10.12989/sss.2023.31.4.365

Deep learning-based post-disaster building inspection with channel-wise attention and semi-supervised learning

Wen Tang [1], Tarutal Ghosh Mondal [2], Rih-Teng Wu*[3], Abhishek Subedi [1] and Mohammad R. Jahanshahi [1,4]

[1] *Lyles School of Civil Engineering*, *Purdue University*, *West Lafayette*, *USA*
[2] *Department of Civil*, *Architecture and Environment Engineering*, *Missouri University of Science and Technology*, *Rolla*, *USA*
[3] *Department of Civil Engineering*, *National Taiwan University*, *Taipei*, *Taiwan*
[4] *Elmore Family School of Electrical and Computer Engineering* (*Courtesy*), *Purdue University*, *West Lafayette*, *USA*

(*Received September 18*, *2022*, *Revised December 17*, *2022*, *Accepted February 2*, *2023*)

Abstract. The existing vision-based techniques for inspection and condition assessment of civil infrastructure are mostly manual and consequently time-consuming, expensive, subjective, and risky. As a viable alternative, researchers in the past resorted to deep learning-based autonomous damage detection algorithms for expedited post-disaster reconnaissance of structures. Although a number of automatic damage detection algorithms have been proposed, the scarcity of labeled training data remains a major concern. To address this issue, this study proposed a semi-supervised learning (SSL) framework based on consistency regularization and cross-supervision. Image data from post-earthquake reconnaissance, that contains cracks, spalling, and exposed rebars are used to evaluate the proposed solution. Experiments are carried out under different data partition protocols, and it is shown that the proposed SSL method can make use of unlabeled images to enhance the segmentation performance when limited amount of ground truth labels are provided. This study also proposes DeepLab-AASPP and modified versions of U-Net++ based on channel-wise attention mechanism to better segment the components and damage areas from images of reinforced concrete buildings. The channel-wise attention mechanism can effectively improve the performance of the network by dynamically scaling the feature maps so that the networks can focus on more informative feature maps in the concatenation layer. The proposed DeepLab-AASPP achieves the best performance on component segmentation and damage state segmentation tasks with mIoU scores of 0.9850 and 0.7032, respectively. For crack, spalling, and rebar segmentation tasks, modified U-Net++ obtains the best performance with Igou scores (excluding the background pixels) of 0.5449, 0.9375, and 0.5018, respectively. The proposed architectures win the second place in IC-SHM2021 competition in all five tasks of Project 2.

Keywords: building visual inspection; channel-wise attention; semantic segmentation; semi-supervised learning

1. Introduction

1.1 Background

Civil infrastructures undergo deterioration over time which necessitates periodic inspection. On the other hand, extreme events like earthquakes induce damage to structures which calls for rapid post-disaster evaluation. However, the current inspection techniques are manual, and therefore, time-consuming, expensive, subjective, and risky. There is a rising need to automate the damage inspection tasks. Recent advances in computer vision techniques have opened lots of opportunities for applications within structural health monitoring (SHM) (Spencer *et al*. 2019, Bao *et al*. 2019, Bao and Li 2021). By leveraging image analysis, the detection of surface defects can be quickly conducted using contactless sensors such as digital cameras, surveillance cameras, or small unmanned aerial vehicles (UAVs). Traditional crack detection approaches are based on techniques including thresholding (Cheng *et al*. 1999, 2003), image percolation (Yamaguchi and Hashimoto 2010), edge detection (Abdel-Qader *et al*. 2003, Fujita and Hamamoto 2011), or morphological operations. However, these techniques are very sensitive to thresholds and need careful manual adjustment to achieve reasonable results. In addition, they tend to work poorly when the background is very noisy.

1.2 Related work

1.2.1 Deep learning based damage detection

To overcome the above-mentioned issues, deep learning provides an alternative solution to damage inspection tasks. Deep learning-based damage detection can be roughly classified into two categories, namely bounding-box-based detection and segmentation. In bounding-box-based detection, most studies perform damage inspection tasks based on region proposal techniques and bounding boxes (Cha *et al*. 2017, 2018, Chen and Jahanshahi 2017, Xue and Li 2018, Maeda *et al*. 2018, Beckman *et al*. 2019, Ghosh Mondal *et al*. 2020). Although these methods can provide a collection of bounding boxes to describe the outline of damaged areas, the results usually contain a large portion of

*Corresponding author, Ph.D., Assistant Professor,
E-mail: rihtengwu@ntu.edu.tw

ISSN: 1738-1584 (Print), 1738-1991 (Online)

the background, especially for crack detection tasks. Compared with bounding-box based detection, damage segmentation is a procedure where each pixel in an image is classified as damage or non-damage. A Fully Convolutional Network (FCN) (Long *et al*. 2015) is proposed to tackle the segmentation task where it uses an encoder-decoder architecture to extract the features and then reconstructs the feature maps to get the final segmentation mask. Inspired by FCN, U-Net (Ronneberger *et al*. 2015) improves the performance of FCN on medical image segmentation by adding extra skip connections between down-sampling paths and the up-sampling paths. Similarly, DeepLabv3+ (Chen *et al*. 2018) model adds spatial pyramid pooling modules and achieves State-Of-The-Art (SOTA) performance on segmentation tasks. Inspired by the above-mentioned segmentation architectures, recent studies (Liu *et al*. 2019, Ji *et al*. 2020, Dung 2019, Zhang *et al*. 2019, Yang *et al*. 2018, Choi and Cha 2019) modify and apply these architectures in damage segmentation and show promising potential for damage segmentation. However, in most of the studies, the images are usually well captured from a predetermined viewpoint on a surface. Other works (Lau *et al*. 2020, Dong *et al*. 2021, Yang and Ji 2021, Liu *et al*. 2020) tried to use U-Net based architectures to segment the crack on concrete surfaces, but the methods are all evaluated on crack images that are shot in close range with relatively small input sizes compared with full-HD images (1920×1080). Moreover, the methods are all tested on images that contain damage, meaning that the test data distribution is skewed towards the damage pixels because not all the collected images contain damage during post-disaster reconnaissances. In Tang *et al*. (2022), the authors proposed to use a two-stage network with Bayesian fusion to segment cracks and reduce the False Positives (FP) cracks using high-resolution images as input. Inspired by the success in Tang *et al*. (2022), for the detailed damage segmentation, this study breaks the full-HD images into patches and then merges them back together when performing detailed damage inspections on the building surface to avoid potential performance degradation due to down sampling. For component and damage state segmentation, the network takes the entire image and predicts the segmentation masks in an end-to-end fashion.

1.2.2 Deep learning based structural component identification

Compared to damage detection and segmentation, less work is done on structural component detection and segmentation. Structural component segmentation is of crucial importance when performing post-disaster damage inspections as the damage of crucial structural components needs more attention than the non-structural components. To perform vision-based bridge component recognition task, the authors in Narazaki *et al*. (2019) applied semantic segmentation networks such as FCN and SegNet to segment the structural components and measured their displacement. In Sajedi and Liang (2021), the authors utilized a U-Net based network and uncertainty maps to segment the structural components. In Gao and Mosalam (2018), the authors use a classification network based on VGG16 to classify the major component inside an image. However, formulating the component identification as a classification task has its natural defects as multiple components might appear in a single image. While existing literature shows promising results using deep learning based techniques for component detection and segmentation, the algorithms could be improved to achieve better performance.

1.2.3 Data issue of post disaster reconnaissance

To enable post-disaster building inspection using UAVs, we need to evaluate the effectiveness of the damage segmentation methods on damaged buildings with images taken from the UAV. Previous studies have indicated that the scarcity of labeled datasets is a major problem in this important area of research. Labeling images for semantic segmentation are an arduous and labor-intensive task. However, this problem can be averted by using synthetic data to train neural network models (Hoskere *et al*. 2022, Narazaki *et al*. 2021, Hoskere et al. 2020). The authors in (Narazaki *et al*. 2021) leverage the synthetic environments to develop a system to segment the structural components of high-speed railway viaducts and further detect and identify the damage of the structural components. Similarly, the authors in (Hoskere *et al*. 2022) built a synthetic environment using Physics-Based Graphics Models (PBGMs) and provide a synthetic dataset for post-disaster building reconnaissance. In Hoskere *et al*. (2022), the authors showed that a neural network trained on synthetic data can also perform reasonably well on real data. Photo-realistic synthetic data generated by computer graphics software can be annotated automatically which saves a lot of time and effort which otherwise goes into the manual annotation of the data. Therefore, the use of synthetic data for the training of neural networks is emerging quickly in the structural health monitoring community. In this study, a synthetic dataset similar to Hoskere *et al*. (2022) is provided by the IC-SHM2021 competition, and the segmentation models are trained and evaluated on this dataset.

1.2.4 Semi-supervised semantic segmentation

Another solution to alleviate the issues brought by the scarcity of labeled datasets is to use SSL. Compared to acquiring the semantic segmentation ground truth of an image, acquiring an image itself without ground truth is much easier. SSL uses both labeled and unlabeled data to train a model, where the labeled data act as a sanity check to establish a base prediction, and the unlabeled data can be used to learn a larger data distribution to improve the performance without having to label all the data. There are many existing literatures on SSL, and consistency regularization is widely adopted in semi-supervised semantic segmentation tasks. The consistency regularization based SSL states that the randomness within the network or other source of perturbations (e.g., data augmentation) should not modify the model predictions given the same input. Therefore, given the same unlabeled image, the predicted masks on the unlabeled image from different networks should be similar. This similarity can be used as supervision signals to train the network on unlabeled images. Among the SOTA of consistency-based SSL methods, PseudoSeg (Zou *et al*. 2020) utilizes data augmentation training and pseudo labeling strategy to

generate the pseudo labels. Cross-consistency training (Ouali *et al*. 2020) applies feature perturbation methods and enforces the consistency between the outputs from perturbed and non-perturbed features. Other methods (Ke *et al*. 2020) utilize network perturbation, where two networks of the same architectures but with different initializations are used and consistency of the outputs of two networks is enforced.

In this study, we incorporate SSL techniques based on consistency regularization and cross supervision, where two networks with the same architecture but different initializations are used. When provided with the same input image, consistency between the output segmentation maps from the two networks is enforced through Cross Entropy (CE) loss.

1.2.5 Channel-wise attention mechanism

In conventional Convolutional Neural Network (CNN) architectures, the channels are the result of convolution filters operating on different features from the input. However, the channels might not have the same representative importance, and conventional CNN does not take the relationship between channels into consideration. Channel-wise attention mechanism (Hu *et al*. 2018) models the importance of different channels by applying different weights to the channels based on their importance before propagating to the next layer, which gives the advantage over conventional CNN. This study takes such channel-wise attention mechanism to modify the state-of-the-art segmentation models for semantic labeling of various structural components and defect categories. In addition, comprehensive experiments are conducted to evaluate the effectiveness of different loss functions and different workflows.

1.2.6 Contributions

First, this study proposes two modified networks for structural component segmentation and damage segmentation based on DeeplabV3+ and U-Net++. The proposed networks utilize channel-wise attention mechanism to rescale the feature maps on each channel according to their importance. Superior performances are achieved when the proposed architectures are compared against their vanilla counterparts. Second, an SSL framework based on consistency regularization is proposed to improve the performance of the model under the assumption that partial labeled data is available. Third, different loss functions and their effect on the performance of the models are discussed and the best loss functions are selected for semantic segmentation tasks.

The remainder of this paper is organized as follows. Section 2 describes how the training, validation, and test data are prepared. Section 3 introduces the framework and methods used in this study. Section 4 presents the analysis and discussions of experimental results. The concluding remarks are addressed in Section 5.

2. Dataset preparation

There are 5 tasks in total in Project 2 of the International Competition for Structural Health Monitoring (IC-SHM2021) competition, namely, component segmentation, damage state segmentation, crack segmentation, spalling segmentation, and rebar exposure segmentation. A total of 4,688 rendered images are obtained from a simulated UAV survey provided by the IC-SHM2021 committee. Out of the 4,688 images, 3,684 images are provided with ground truth labels and the rest 1,004 images are test images without ground truth labels. In addition, we draw 500 images out of the 3,684 images to construct our local test data. In order to train and evaluate the model without overfitting, the rest of the 3,184 labeled images (without local test images) of size 1920×1080 are split into training and validation sets with a ratio of 87.5% and 12.5% respectively.

For the detailed damage segmentation task, including crack, spalling, and rebar exposure segmentation tasks, the 3,184 labeled images are cropped into image patches in a sliding window fashion with a step size of 200 pixels in height and width direction. Considering the imbalance of background and foreground pixels in damage segmentation tasks, the image patches without the foreground objects (crack, rebar exposure, and spalling) are removed. For crack, spalling, and rebar exposure segmentation tasks, we generate a different dataset for each of the tasks. Therefore, there are 38,021 training and validation image patches for crack segmentation, 39,502 image patches for spalling segmentation, and 15,654 image patches for the rebar exposure segmentation task.

For component segmentation and component damage state task, we feed the full resolution images into the segmentation network without cropping the images.

To enrich the dataset and avoid overfitting, a series of data augmentation techniques including flipping, shifting, random contrast, random brightness, random hue shifting, and random scaling are used. It is noted that these image augmentations are made on the fly while training, so the augmented images are not counted towards the total number of images or image patches in the dataset.

For SSL, three different data partition cases are considered. Each of these cases uses a different partition ratio and the performance of the segmentation networks is evaluated based on sub-sampling of the datasets with a ratio $1/n$ for the labeled set and $(1 - 1/n)$ for the unlabeled set. Specifically, in the component and damage state segmentation, the training and validation images are combined (3,184 images) and split with the ratios of 1/8, 1/4, 1/2 for the labeled set. The rest of the 7/8, 3/4, and 1/2 images are treated as unlabeled images. Similarly, for crack, spalling, and rebar segmentation tasks, the training and validation patches are combined and split with the ratios of 1/8, 1/4, 1/2 for the labeled set. The test scores are reported on the same 500 test images.

3. Methodology

3.1 Overall workflow

In this study, we propose two different segmentation workflows using four segmentation models, including vanilla U-Net++, vanilla DeeplabV3+, modified U-Net++, and DeepLab-AASPP to segment the structural components

and the detailed damage segmentation. The detailed architectures are introduced in Sections 3.2 and 3.3.

For structural component and damage state segmentation, we propose DeepLab-AASPP which is an attention-based DeepLabv3+ extension. The proposed network utilizes the attention mechanism to focus more on the informative feature maps in the ASPP module of DeepLabV3+. The network takes the entire image and predicts the segmentation masks in an end-to-end fashion.

For detailed damage segmentation, we propose a modified architecture based on U-Net++ (Zhou *et al*. 2018) to segment out the damaged part. A pixel-level ensembling method is proposed to aggregate the information from overlapping patches and the results turn out to be much more robust than the baseline model without ensembling. During the inference stage, the entire 1920×1080 image is first cropped into overlapped image patches using a sliding window of step size 100. Then each image patch is fed into the segmentation network to predict the mask. Finally, the predicted segmentation masks based on cropped image patches are merged back together to form the final damage segmentation mask for the entire image. The average of SoftMax scores of the same pixel presented in different image patches is used to determine the final label of that pixel.

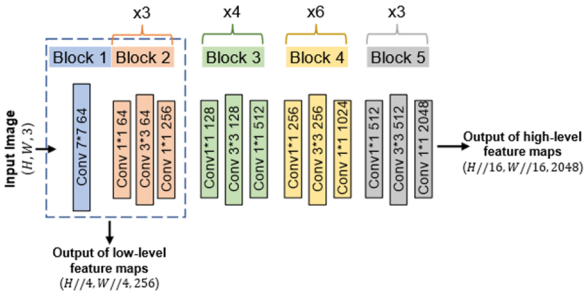


Fig. 1 Architecture of ResNet101 as the backbone. The skip connections are ignored for simplicity

3.2 DeepLab-AASPP

3.2.1 Encoder

In order to better extract the features of input images, we use pre-trained ResNet (He *et al*. 2016) as the encoder of the proposed network, which is referred to as the "backbone" architecture shown in Fig. 1. There are a series of ResNet models ranging from ResNet34 to ResNet152 where a larger number indicates deeper and more complex architecture. Considering the computational memory cost and the relative performance of the ImageNet classification Dataset, we pick ResNet101 as the backbone structure of the proposed Lab-AASPP network. The encoder has 2 output branches, one is the low-level feature branch and the other one is the high-level feature branch.

As shown in Fig. 1, the low-level feature branch is extracted from the earlier stage of ResNet101 which is used to perverse some low-level features extracted from the backbone structure. On the other hand, the high-level feature maps are from the output of "Block 5" where the feature maps are down sampled 16 times compared with the input dimension. The output feature maps from the high-level feature branch are then fed into the AASPP module, whereas the low-level feature maps are fed into the decoder part.

3.2.2 Attention-ASPP

Different from most encoder-decoder like FCN, the original DeepLab model family provides an architecture for learning using multi-scale contextual features through the ASPP module. ASPP in DeepLab uses several grid scales to extract features at different scales and resolutions. The ASPP is different from the conventional Spatial Pyramid Pooling (SPP) (He *et al*. 2015) in the sense that atrous convolutions are used instead of the traditional convolution operation. Atrous convolutions are conventional convolutions with zeros values inserted between the kernels' values. It is used to increase the receptive field of the convolutional kernels and control the spatial resolution of the feature maps. Since only zero values are added, the number of learnable parameters stays unchanged, and the computational cost is about the same. The equation of atrous convolution can be expressed as follow

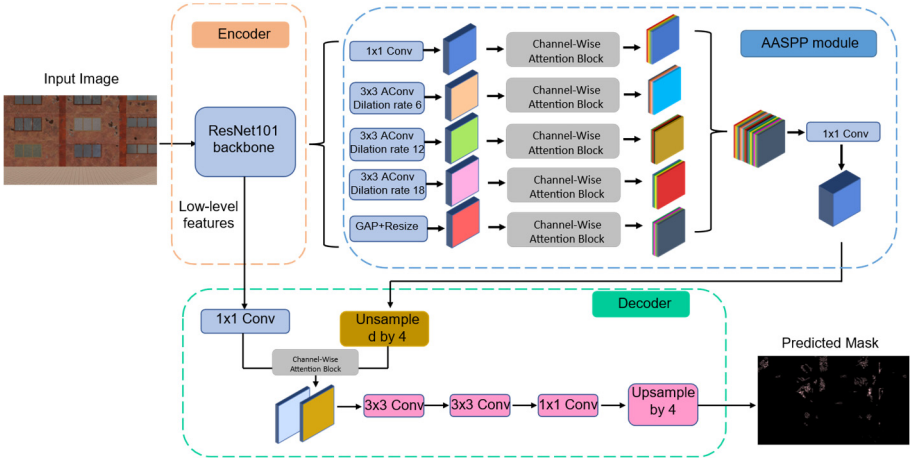


Fig. 2 Detailed architecture of the DeepLab-AASPP segmentation network

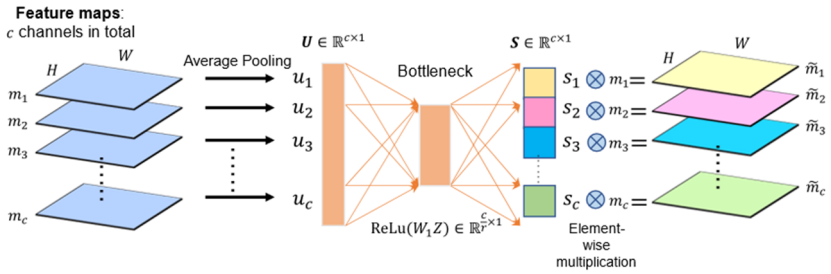


Fig. 3 Schematic diagram of Attention Atrous Spatial Pooling Pyramid (AASPP)

$$y[m,n] = \sum_j \sum_k x[m - r \cdot j, n - r \cdot k]\, w[j,k] \quad (1)$$

where x is the input feature map, y is the output feature map, w is the convolution kernel, (m,n) is the spatial location on the output feature map y, and r is the dilation rate which is the number of zeros inserted along the height and width directions of the convolution kernels. As can be seen in Fig. 2, there are 4 atrous convolution layers with dilation rates of 1, 6, 12, and 18 respectively. And there is an additional global average pooling (GAP) layer used parallel to the 4 atrous convolution layers.

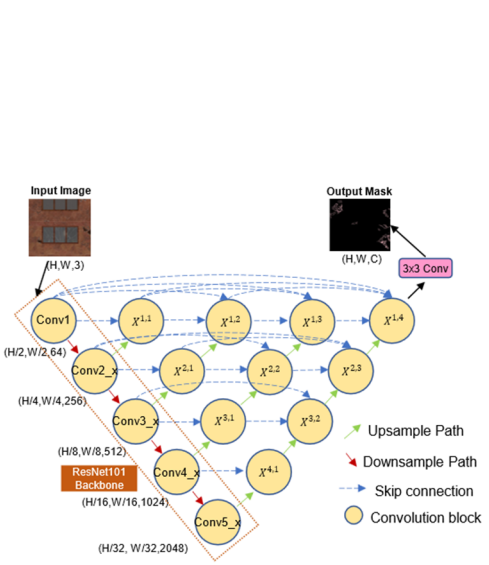
The original ASPP module in DeepLabV3+ concatenates all the output feature maps from the above-mentioned 5 layers and feeds them to the decoder module of the network. The feature maps are processed with equal attention while some of them could be more informative than others. Inspired by Squeeze-and-Excitation (Hu *et al.* 2018), the proposed Attention-ASPP module is designed to improve the representational power of the original ASPP module by adding channel-wise attention to the output feature maps. It helps the network focus on more informative feature maps by weighting each feature map differently using a unique weight for each channel. In Fig. 3, the global context of the feature maps is first extracted through a global average pooling layer expressed as follows

$$u_i = \frac{1}{H \times W} \sum_j^H \sum_k^W m_i(j,k) \quad (2)$$

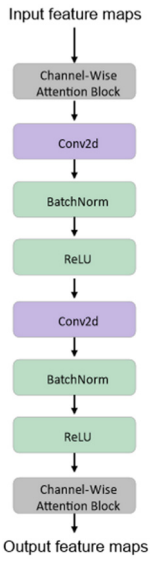
where, u_i is the output of the global average pooling layer, H and W are the height and width of the input feature map m_i, j and k are the spatial locations of the feature in height and width direction. After that, $\boldsymbol{U} \in \mathbb{R}^{c \times 1}$ is squeezed into $c//r$ number of neurons using a dense layer followed by ReLU activation. Finally, another dense layer composed of c neurons followed by sigmoid activation is used to calculate the weight of each channel. The final weighted feature map $\tilde{m}_i$ are obtained by scaling each feature map m_i with the corresponding weight s_i.

3.3 Modified U-Net++

Unet++ is a powerful encoder-decoder network based on U-Net. Similar to U-Net, U-Net++ uses down-sampling paths to extract and encode the information from the image, and up-sampling paths to scale up the generated feature maps and generate the segmentation masks. However, U-Net++ utilizes a series of short and long dense skips to fight against gradient vanishing and improve the gradient flow.



(a) Schematic diagram of modified U-Net++



(b) Convolution block in modified U-Net++

Fig. 4 Detailed architecture of the modified U-Net++

Besides that, there are convolution layers on skip connections to bridge the semantic gap between the encoder and decoder feature maps.

In this study, we modified the U-Net++ by embedding the channel-wise attention in the convolution block $X^{i,j}$, shown in Fig. 4(a). As shown in Fig. 4(b), for each convolutional block, the channel-wise attention blocks are added after the input feature maps and the output feature maps from the last ReLU layer. For other configurations of the channel-wise attention blocks, readers can refer to Table 7 for more details. These embedded attention blocks are the same as the channel-wise attention blocks in the Attention-ASPP module. The goal of these attention blocks is to weigh different feature maps and pay more attention to the informative feature maps. Similar to DeepLab-AASPP, ResNet101 is adopted as the backbone architecture to encode the information from raw pixels.

3.4 Loss function evaluation

We also consider the influence of different loss functions to mitigate the imbalance data distribution issue, especially for high imbalance datasets like crack, spalling, and rebar segmentation tasks. In this study, 5 different loss functions are considered including Cross Entropy (CE) loss, Weighted Cross Entropy (WCE) loss, Focal loss, Lovasz loss, and Lovasz-WCE loss. Detailed information is given in the following sections.

3.4.1 Cross-entropy loss functions

Cross Entropy loss builds upon the idea of entropy and is one of the most commonly used loss functions in all sorts of machine learning tasks. In the segmentation task, the loss function is calculated by a pixel-wise SoftMax over the final feature maps. This loss function examines each pixel individually and compares the class prediction with the one-hot-encoded vector in the ground truth mask. The equation for Cross Entropy loss is expressed as follows

$$\mathcal{L}_{CE} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{c=0}^{C-1} y_{true}^{i,c}\, log\,(\hat{p}_y^{i,c}) \tag{3}$$

where N is the total number of pixels in the ground truth mask, C is the total number of classes, y_{true} is the one-hot-encoding vector for the ground truth label, and $\hat{p}_y$ is the predicted vector whose elements sum up to one. Because Cross Entropy loss is averaged over all the pixels, the network essentially pays equal attention to all the pixels in the image. This could be problematic in an unbalanced dataset since the learning could be dominated by the pixels belonging to the major class whereas the less frequent class might be neglected. This phenomenon is undesirable in tasks like damage segmentation and could lead to "all-background" predictions where damage pixels clearly belong to the minority class. Weighted Cross Entropy $\mathcal{L}_{WCE}$ is adapted to mitigate this issue by assigning different weights to different classes when calculating the loss function. By assigning larger weights to underrepresented classes, the network pays more attention to the minor classes during the learning process. The weights w_c is calculated using bounded inverse class weight (Paszke et al. 2016) which is defined as: $w_c = 1/\ln(\beta + p_c)$ where $p_c = \frac{num.\ pixels\ in\ class\ c}{Total\ num.of\ pixels}$ is the probability of a pixel belonging to class c, and β is a constant that is set to 1.02 to bound the weight between the interval of [1,50] so that the class weight is not too big to mess up the gradient update.

$$\mathcal{L}_{WCE} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{c=0}^{C-1} w_c * y_{true}^{i,c}\, log\,(\hat{p}_y^{i,c}) \tag{4}$$

3.4.2 Focal loss

Focal loss (Lin *et al*. 2017) is also proposed to mitigate the unbalance issue by applying the modulating term to the CE loss function. It dynamically scales the CE loss and anneals the scaling factor as the confidence in the correct class increases. In the multi-class setting, with integer labels y, Focal loss is defined as

$$\mathcal{L}_{focal}\left(y_{true}, \widehat{\boldsymbol{P}}\right) = -\left(1-\hat{p}_{y_{true}}\right)^{\gamma}\log\left(\hat{p}_{y_{true}}\right) \tag{5}$$

where, $y_{true} \in \{0, \dots, C-1\}$ is an integer class label with C denotes the number of classes, $\widehat{\boldsymbol{P}} = (\hat{p}_0, \hat{p}_1, \dots, \hat{p}_{c-1}) \in [0,1]^C$ is the predicted vector with each element being the probability of that class, and γ is the parameter that controls how much higher-confidence correct predictions contribute to the overall loss. As γ gets higher, the correct high-confidence predictions contribute less to the total loss. And the ordinary multi-class Cross Entropy loss will be recovered if $\gamma = 0$. In this study, we set $\gamma = 2$ as suggested in Lin *et al*. (2017).

3.4.3 Lovasz loss

The Lovasz-Softmax loss, originally proposed by Berman *et al*. (2018), is a loss function for multi-class segmentation that incorporates the SoftMax operation in Lovasz extension. This loss directly optimizes towards the mIoU because it serves as a differentiable surrogate function of the Jaccard coefficient. It is relatively difficult to completely define the function in an equation as it is an optimization algorithm based on sorted errors. We refer the reader to Berman *et al*. (2018) and Bermanmaxim (2018) for detailed information and implementation.

Berman *et al*. (2018) claim that the Lovasz loss can be optimized automatically, but the finetuning of parameters depends on the batch size used and the number of classes in the dataset. To better fine-tune the parameters, it is suggested to train the network with Cross Entropy loss then switch to Lovasz loss, or combine the Cross-Entropy loss with Lovasz loss. In this study, we combine the Weighted Cross Entropy loss with Lovasz loss with addition. The definition of the combined loss is shown as follows, where $\alpha \in (0,1)$ is a real number between 0 and 1 that controls the mixture ratio between these two loss functions. In this study, we set $\alpha = 0.5$.

$$\mathcal{L}_{LWCE} = \alpha\mathcal{L}_{WCE} + (1-\alpha)\mathcal{L}_{Lovasz} \tag{6}$$

3.5 Consistency based semi-supervised learning

Inspired by Chen *et al*. (2021) and Hu *et al*. (2018), an SSL framework based on cross consistency of pseudo labels is proposed to train the networks using both labeled and unlabeled data. The proposed approach imposes consistency between the predictions from two segmentation networks with different initialization given the same input image. The output pseudo one-hot encoding segmentation map from one segmentation network is used to supervise the training of the other segmentation network, and vice versa. Cross Entropy loss is used to enforce consistency, and backpropagation is performed on both networks in this approach.

Given a labeled dataset $\mathcal{D}^l = \{(\mathbf{x}_i, \boldsymbol{y}_i)\}_{i=1}^{|\mathcal{D}^l|}$ and an unlabeled dataset $\mathcal{D}^u = \{(\mathbf{x}_i, \boldsymbol{y}_i)\}_{i=1}^{|\mathcal{D}^u|}$. Input image $\mathbf{x}_i \in \boldsymbol{\chi} \subset \mathbb{R}^{H\times W\times K}$ of size $H \times W$ with K color channels is fed into two parallel segmentation networks f_1 and f_2 with different initialization of parameters $\boldsymbol{\theta}_1$ and $\boldsymbol{\theta}_2$

$$\boldsymbol{p}_1^i = f_1(\mathbf{x}_i; \boldsymbol{\theta}_1) \tag{7}$$

$$\boldsymbol{p}_2^i = f_2(\mathbf{x}_i; \boldsymbol{\theta}_2) \tag{8}$$

where $\boldsymbol{p}_1^i$ and $\boldsymbol{p}_2^i$ are the segmentation masks with SoftMax score for each pixel. The corresponding one-hot label masks derived from $\boldsymbol{p}_1^i$ and $\boldsymbol{p}_2^i$ are denoted as $\widehat{\boldsymbol{y}}_1^i$ and $\widehat{\boldsymbol{y}}_2^i$.

The loss function of the SSL consists of two parts: the supervised learning loss $\mathcal{L}_s$ (Eq. (9)) and the unsupervised learning loss $\mathcal{L}_u$ (Eq. (10)). The supervised loss $\mathcal{L}_s$ is formulated using the Lovasz-WCE loss $\mathcal{L}_{Lovasz-WCE}$ as mentioned in Section 3.4, and it is only calculated on labeled images.

$$\mathcal{L}_s = \frac{1}{|\mathcal{D}^l|}\sum_{X\in\mathcal{D}^l}\frac{1}{W\times H}\sum_{i,j=0,0}^{W-1,H-1}\left(\mathcal{L}_{LWCE}\left(\boldsymbol{p}_1^{i,j}, \boldsymbol{y}_{true}^{i,j}\right) + \mathcal{L}_{LWCE}\left(\boldsymbol{p}_2^{i,j}, \boldsymbol{y}_{true}^{i,j}\right)\right) \tag{9}$$

For the unsupervised loss, the one-hot encoding mask $\widehat{\boldsymbol{y}}_1^i$ is used to supervise the pixel-wise SoftMax score mask$\boldsymbol{p}_2^i$, which is the output of network $f_2(\boldsymbol{\theta}_2)$. Similarly, the one-hot encoding mask $\widehat{\boldsymbol{y}}_2^i$ is used to supervise the pixel-wise SoftMax score mask $\boldsymbol{p}_1^i$, which is the output of network $f_1(\boldsymbol{\theta}_1)$. Unlike the supervised loss, the unsupervised loss $\mathcal{L}_u$ is calculated on both the labeled and unlabeled dataset. The definition of the unsupervised loss $\mathcal{L}_u$ can be expressed as follows

$$\mathcal{L}_u = \frac{1}{|\mathcal{D}^u+\mathcal{D}^l|}\sum_{X\in\{\mathcal{D}^u,\mathcal{D}^l\}}\frac{1}{W\times H}\sum_{i,j=0,0}^{W-1,H-1}\left(\mathcal{L}_{CE}\left(\boldsymbol{p}_1^{i,j}, \widehat{\boldsymbol{y}}_2^{i,j}\right) + \mathcal{L}_{CE}\left(\boldsymbol{p}_2^{i,j}, \widehat{\boldsymbol{y}}_1^{i,j}\right)\right) \tag{10}$$

The entire loss function (Eq. (11)) for the proposed SSL framework is the combination of supervised loss and unsupervised loss, which can be expressed as follows

$$\mathcal{L}_{semi} = \mathcal{L}_s + \mathcal{L}_u \tag{11}$$

The intuition is that the cross supervision, using $\widehat{\boldsymbol{y}}_2$ to supervise $\boldsymbol{p}_1$ and $\widehat{\boldsymbol{y}}_1$ to supervise $\boldsymbol{p}_2$, can enforce the prediction consistency between the two models, while the supervised loss $\mathcal{L}_s$ on labeled data can guide both models towards the right direction. This way, we can utilize the extra images in the dataset even if the ground truth segmentation mask is not available.

3.6 Training details

The proposed networks are trained on a computation platform including a Linux server that consists of two Intel Xeon E5-2620 CPUs, 256 GB DDR4 RAM and eight NVIDIA RTX Quadro 8000 GPUs with 48 GB memory. As for the software, we use PyTorch as our implementation platform.

For components segmentation and component damage state estimation, we use the full resolution image without any preprocessing as the input. For crack, rebar and spalling damage segmentation tasks, we are cropping the original

Table 1 Summary of training parameters

Hyper parameters	Component and damage state segmentation	Detailed damage segmentation (crack, spall and rebar)
Input size	1920×1080×3	512×512×3
Sliding window step size	N/A	200 (training)/100 (inference)
Optimizer	Adam	
Base learning rate (LR_{base})	1e-4	
LR decay function	$LR = LR_{base}\left(1 - \frac{Iter_{current}}{Iter_{max}}\right)^{0.9}$	
Max iterations ($Iter_{max}$)	40,287 iterations (DeepLab) / 86,030 iterations (U-Net++)	
Pretrained checkpoint	ImageNet	
Batch size	8(DeepLab)/12(U-Net++)	

images into image patches of 512×512 to avoid the detail loss caused by resizing. To avoid overfitting on the training set, we use heavy image augmentations including flipping, scaling, shifting, random contrast, random hue shift, and random brightness changing.

In this study, we use transfer learning method to train our models where the initial weights of the encoder backbones are from ImageNet (Deng *et al*. 2009). The input images are also normalized in an ImageNet fashion. Adam optimizer is utilized with a starting learning rate of 1e-4. An exponential decaying learning rate scheduler is used to gradually decrease the learning rate to better find the global minimum. For batch size, we keep the batch size as large as possible until the GPU goes out of memory. Table 1 is the summary of the learning parameter used in this study.

To have a fair comparison between the supervised and SSL, we follow the training strategy from (Ke *et al*. 2020) where the number of training iterations is approximately the same for both supervised learning and SSL. For SSL, half of the images in the mini-batch are labeled images and the other half of the images are unlabeled images. To be more specific, for component segmentation task under 1/8 data partition, 398 images are considered as labeled images in the labeled set, and 2,786 images are considered as unlabeled images without the ground truth mask. In the case of SSL, 4 images from the labeled dataset, and 4 images from the unlabeled dataset are sampled at each iteration to form a mini-batch and fed into the networks for training. Whereas in the case of supervised learning baseline, 8 images are sampled from the labeled dataset to form a mini-batch at each iteration, and the unlabeled dataset is not used.

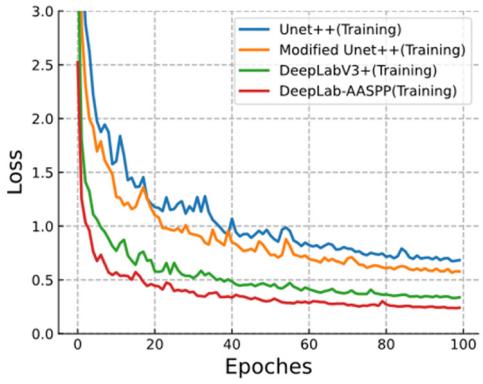
4. Results and discussions

The performance of the proposed networks along with different loss functions are reported in this section. It should be noted that IoU from damage class is used in binary segmentation tasks like crack, spalling and rebar segmentation tasks, and mIoU is used for multi-class segmentation tasks like component and damage state segmentation tasks. The definition of $mIoU$ score is given below

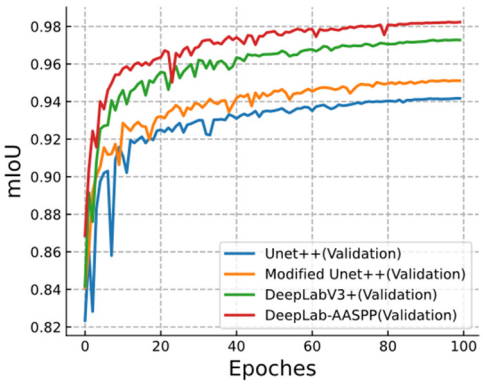
$$mIoU = \frac{1}{N_{class}} \sum_{i=1}^{N_{class}} \frac{TP_i}{TP_i + FP_i + FN_i} \tag{12}$$

$$IoU = \frac{TP_{damage}}{TP_{damage} + FP_{damage} + FN_{damage}} \tag{13}$$

The models are evaluated on 5 tasks, namely component segmentation, component damage state segmentation, crack segmentation, rebar segmentation and spalling segmentation. As mentioned in Section 2, the input of the segmentation network on the detailed damage dataset is cropped image patches of size 512×512. Therefore, the loss and $mIoU$ in Fig. 7 are metrics based on cropped crack images. The $mIoU$ might drop on the full-scale images because of potential false positives on pure background image patches.



(a) Training loss curve of proposed networks



(b) Validation mIoU curve of proposed networks

Fig. 5 Training loss and validation mIoU of the proposed networks in component segmentation task when Lovasz-WCE loss function is used

Table 2 mIoU score of different architectures and loss functions in component segmentation task

Architectures	Loss functions					Parameters (Millions)
	CE	WCE	Focal	Lovasz	Lovasz-WCE	
Vanilla U-Net++	0.9189	0.9217	0.9221	0.9428	0.9460	67.99
Modified U-Net++	0.9272	0.9312	0.9311	0.9524	0.9562	68.94
DeepLabV3+	0.9426	0.9451	0.9449	0.9612	0.9633	59.34
DeepLab-AASPP	0.9534	0.9576	0.9578	0.9810	0.9850	59.36

4.1 Results of different tasks

4.1.1 Component segmentation task

The training curves of the considered four models are plotted in Fig. 5 to compare the convergence and loss values of different models on different datasets. It can be observed that the training loss of DeepLab based models decrease faster and converged at lower loss levels compared with U-Net based models. The highest validation mIoU of 0.9850 is achieved with the DeepLab-AASPP model (modified DeepLab).

From Table 2, it can be observed that the improvements induced by channel-wise attention mechanism when introduced into both architectures. Modified U-Net++ with channel-wise attention outperforms its counterpart Vanilla U-Net++ by 1.1%. DeepLab with channel-wise attention in the ASPP module is also superior to vanilla DeepLabv3+ model with an improvement of 2.17% on mIoU. In addition, Table 2 shows that for the same architecture, introducing Lovasz loss can boost the mIoU performance by 3.0% on average when compared with CE loss.

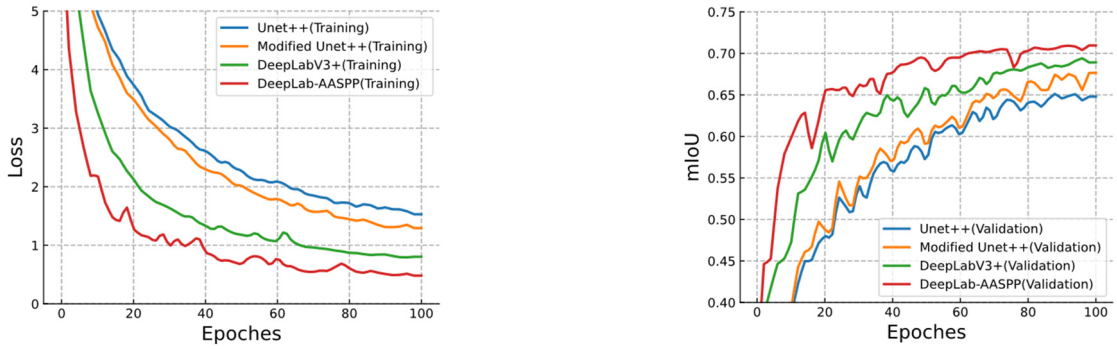
4.1.2 Damage state segmentation task

The training and validation curves of the four models are plotted in Fig. 6. Similar to component segmentation task, it can be observed that the best test mIoU of 0.7032 is achieved when DeepLab-AASPP model is used. In damage state segmentation task, the final mIoU increased 1.31% and 2.30% on 500 local test images when including channel-wise attention in Vanilla U-Net++ and DeepLabv3+. From Table 3, it can be observed that the Lovasz-WCE loss can effectively improve the mIoU when compared with other loss functions.

4.1.3 Detailed damage segmentation tasks

For detailed damage segmentation tasks, we perform a heuristic model search to find the best model for detailed damage segmentation. Since crack, rebar and spalling damage segmentation share similar workflow and nature of data, we assume that the optimal loss function on crack segmentation would also have a good performance on rebar and spalling segmentation tasks. In addition, it is intractable to consider all possible combinations of hyper-parameters and training settings. It should be noted that the workflow of detailed damage segmentation is different from component segmentation. To preserve the damage details as much as possible and keep a reasonable input size, the full-scale image is first cropped into overlapping patches. Then the damage segmentation is performed on each image patch. Finally, the image patches are stitched back and registered on the full-scale image using average SoftMax damage score of each pixel in the overlapping area.

Training curves of the four models using Lovasz-WCE loss on crack damage segmentation dataset are given in Fig. 7. It can be observed that the training loss of U-Net++ based models decrease faster and converge at lower loss levels compared with DeepLab based models. The highest test IoU of 0.5449 is achieved with modified U-Net++ model. This observation is different from what is observed in component and damage state segmentation task, where the U-Net based models perform better than the DeepLab based models. Similar to component and damage state segmentation, the embedding channel-wise attention modules bring improvements to the models regardless of their architectures according to Table 4. Embedding channel-wise attention module brings 2.6% and 2.9% IoU improvements to vanilla U-Net++ and DeepLabv3+ when



(a) Training loss curve of proposed networks

(b) Validation mIoU curve of proposed networks

Fig. 6 Training loss and validation mIoU of proposed networks in damage state segmentation task when Lovasz-WCE loss function is used

Table 3 mIoU score of different architectures and loss functions in damage state segmentation task

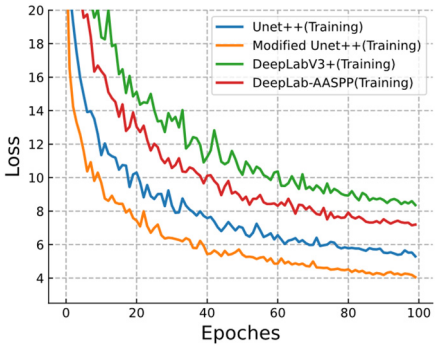
Architectures	Loss functions					Parameters (Millions)
	CE	WCE	Focal	Lovasz	Lovasz-WCE	
Vanilla U-Net++	0.6388	0.6391	0.6390	0.6603	0.6651	67.99
Modified U-Net++	0.6509	0.6517	0.6502	0.6779	0.6782	68.94
DeepLabV3+	0.6601	0.6631	0.6621	0.6789	0.6802	59.34
DeepLab-AASPP	0.6708	0.6742	0.6748	0.7011	**0.7032**	59.36

Table 4 IoU of crack pixels for different architectures and loss functions in crack segmentation

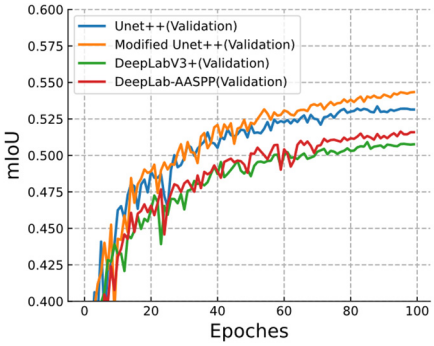
Architectures	Loss functions					Parameters (Millions)
	CE	WCE	Focal	Lovasz	Lovasz-WCE	
Vanilla U-Net++	0.4639	0.4797	0.4784	0.5131	0.5185	67.99
Modified U-Net++	0.4831	0.4999	0.4989	0.5309	**0.5449**	68.94
DeepLabV3+	0.4508	0.4670	0.4669	0.4814	0.4856	59.34
DeepLab-AASPP	0.4631	0.4748	0.4734	0.5052	0.5144	59.36

Table 5 IoU scores of damage pixels for different architectures in spalling and rebar segmentation tasks. All the architectures in this table use the Lovasz-WCE loss function

Tasks	**Architectures**			
	Vanilla U-Net++	Modified U-Net++	DeepLabV3+	DeepLab-AASPP
Rebar	0.4864	**0.5018**	0.4671	0.4703
Spalling	0.9031	**0.9375**	0.8748	0.8954



(a) Training loss curve of proposed networks



(b) Validation mIoU curve of proposed networks

Fig. 7 Training loss and validation IoU (excluding background IoU) of the proposed networks in crack segmentation task when Lovasz-WCE loss function is used

combined with Lovasz-WCE loss.

After experimenting with different losses on the crack dataset, we believe the Lovasz-WCE loss function is the superior loss function when dealing with an extremely unbalanced dataset where the total number of background pixels is significantly larger than the damage pixels. To confirm the optimal models on rebar and spalling segmentation datasets, we trained the models again on rebar and spalling segmentation with the Lovasz-WCE loss function, and the results are given in Table 5. Table 5 shows that the modified U-Net++ model achieves the best test performance on rebar and spalling segmentation tasks, which aligns with the observation in the crack segmentation dataset. For rebar segmentation, modified U-Net++ brings 1.5% on the IoU score compared with vanilla U-Net model, where improvement brought by embedding channel-wise attention modules is smaller compared with other tasks. A similar phenomenon is observed when comparing DeepLabv3+ with DeepLab-AASPP model, where the channel-wise attention only gives a 1.32% gain on IoU score. One explanation is that the extremely unbalanced rebar/background pixel distribution makes it difficult for the models to learn the features of exposed rebars. Lacking training examples (rebar pixels) could severely limit the performance of modified models even if their architectures are superior compared to vanilla models. For spalling segmentation task, the additional channel-wise attention modules bring 3.4% and 3.00% improvements in IoU score compared with vanilla U-Net++ and DeepLabv3+ models.

4.2 Analysis and discussion

4.2.1 Comparison of different architectures

Tables 2, 3, 4, 5 compare the mIoU/IoU performance of different models on 500 full-scale local test images. It can be observed that the DeepLab-ASPP model works the best on component and damage state segmentation tasks, bringing 3.0% and 3.7% improvement in mIoU score when compared with modified U-Net++. In addition, DeepLabv3+ model also outperforms the vanilla U-Net++ model on these two separate tasks. Even though the total parameters of DeepLab based models are less than the U-Net++ based models, the performance of DeepLab based models is still better on component and damage state segmentation tasks. One of the reasons that DeepLab models perform better than the U-Net++ models is the Atrous Spatial Pyramid Pooling module where the feature maps are resampled at multiple rates with multiple parallel

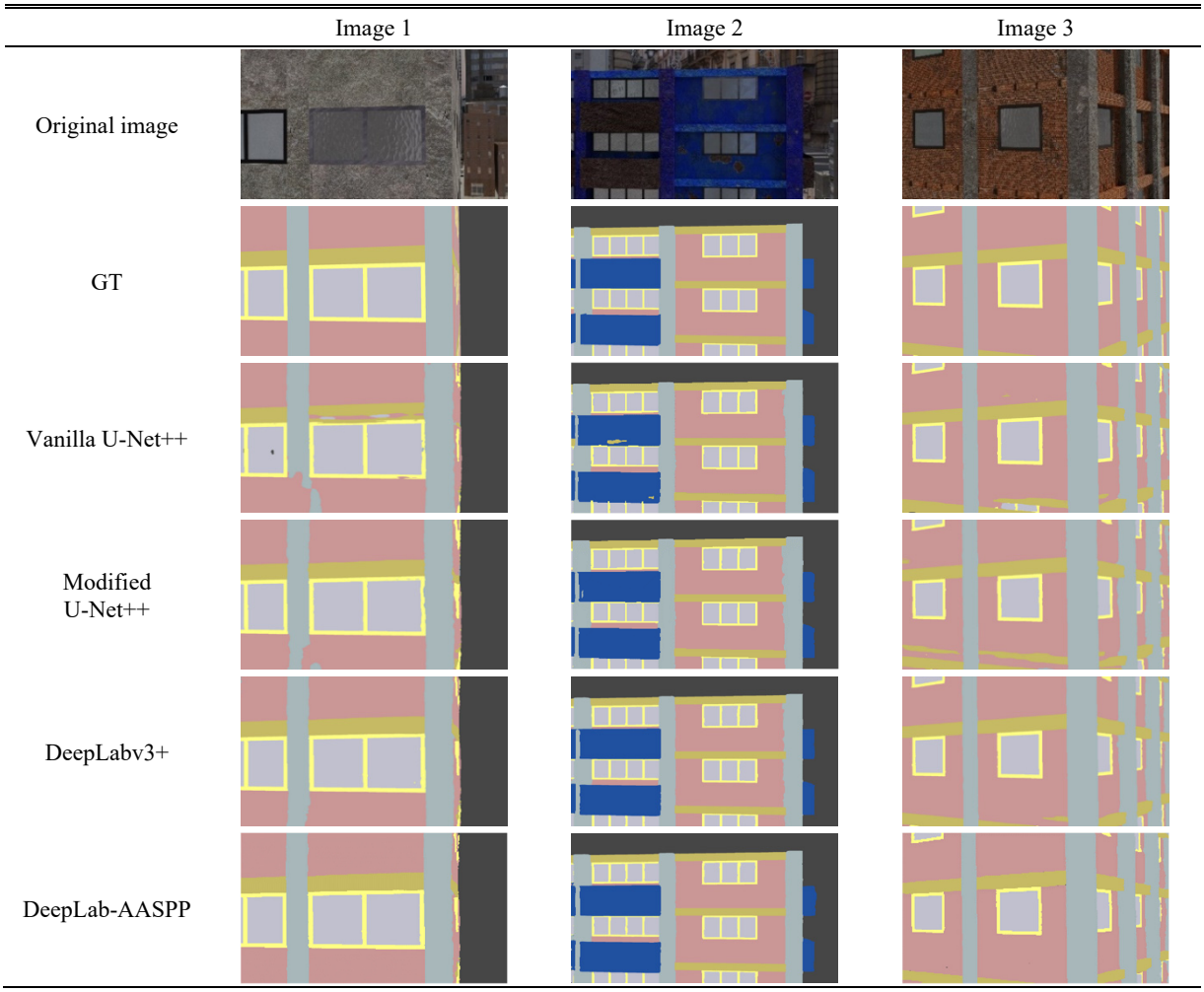


Fig. 8 Sample data and comparison of the predicted component masks of different architectures

atrous kernels. These kernels with complementary effective fields of view can capture features at multiple scales and use global information to improve the mIoU.

However, for crack, rebar and spalling segmentation tasks, modified U-Net++ models work the best and bring 3.1%, 3.2% and 4.2% IoU improvements on these three datasets when compared with DeepLab-ASPP model. This performance improvement might cause by the extra parameters and the denser connections between the layers, which could help the network in predicting more detailed features and improve the IoU score.

Prediction results using different architectures are shown in Figs. 8, 9 and 10. It can be observed that both U-Net++ and DeepLab based architectures can segment the majority of the data quite well despite the cluttered background. In component and damage state segmentation tasks where larger color blocks need to be segmented out, DeepLab based models did better by predicting more complete color blocks and crispier boundaries compared with the U-Net++based models. Moreover, additional channel-wise modules lead to better boundary predictions compared to both vanilla models.

To better compare and illustrate the performance difference in detailed damage segmentation tasks, we compare the performance of different models on zoomed-in image patches in Fig. 10. It can be observed that both DeepLab based models and U-Net++ based models can segment out the major damage area quite well despite of the noisy background. However, a detailed comparison indicates that the U-Net++ based models can predict boundaries that are closer to the ground truth label compared with the DeepLab based models. In addition, the extra channel-wise attention modules further enhance the performance by refining the boundaries and recovering smaller damage areas.

4.2.2 Comparison of different loss functions

According to Tables. 2 and 3, compared with models trained with Cross Entropy loss, DeepLab-AASPP trained with Lovasz-WCE loss increases the mIoU from 0.9534 to 0.9850 on component damage segmentation task, and from 0.6708 to 0.7032 on damage state segmentation. For modified U-Net++ trained on crack, rebar and spalling datasets, switching from CE to Lovasz-WCE loss improves the IoU of damage pixels by 6.18%, 5.10% and 8.60% respectively, according to Figs. 11(c)-(e).

To compare the effects of different loss functions on each class, Fig. 11 plotted the change of IoU of each class with respect to different loss functions in all 5 tasks. It can be observed that the IoU of minority classes ("window

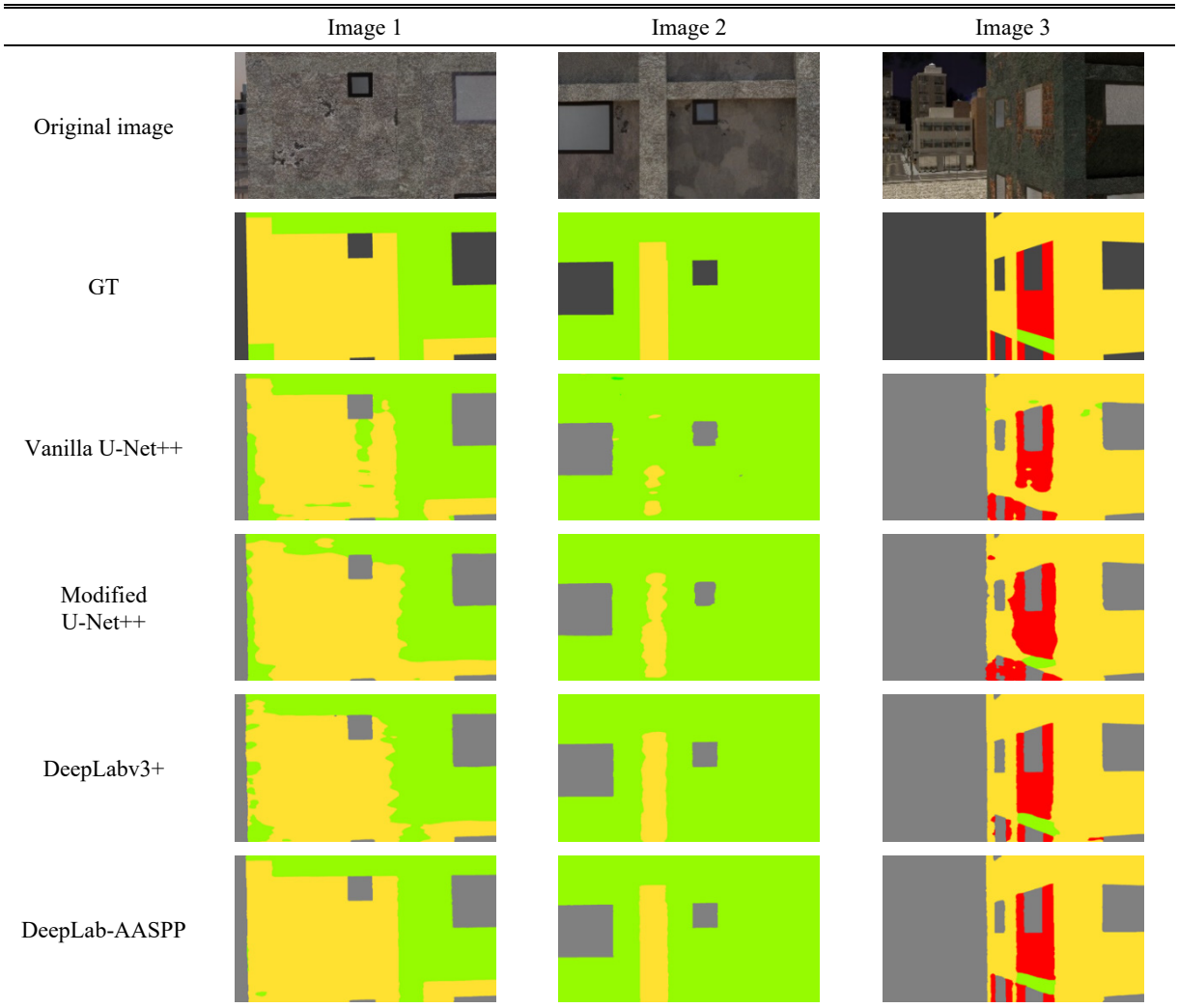


Fig. 9 Sample data and comparison of the predicted damage state masks of different architectures

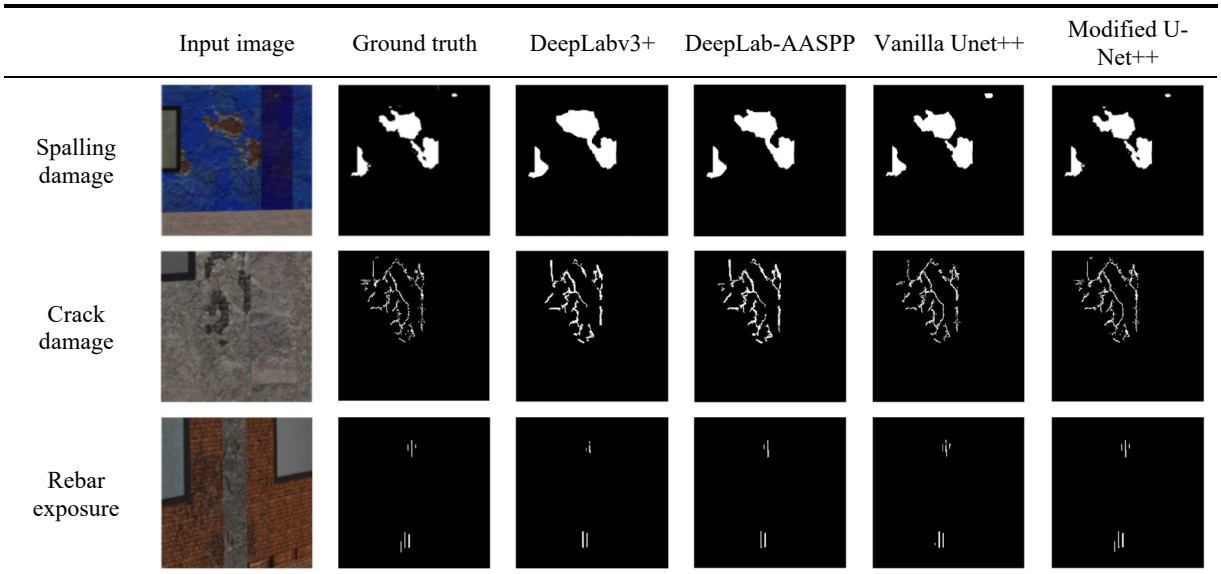
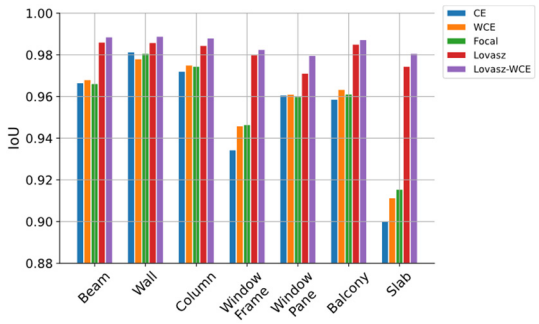
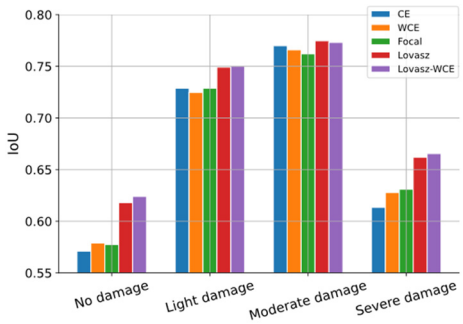


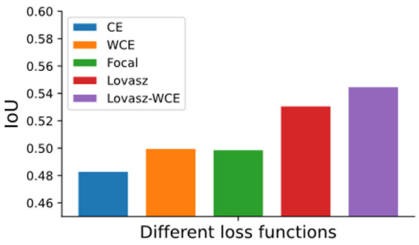
Fig. 10 Sample data and comparison of the predicted damage masks of different architectures in detailed damage segmentation tasks



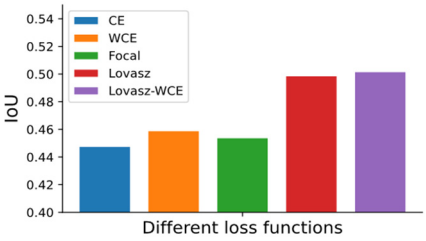
(a) IoU of each class when DeepLab-AASPP trained on component segmentation task



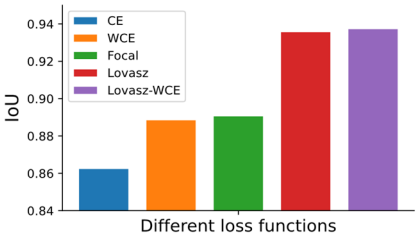
(b) IoU of each class when DeepLab-AASPP trained on damage state segmentation task



(c) IoU of crack pixels of Modified U-Net++



(d) IoU of rebar pixels of Modified U-Net++



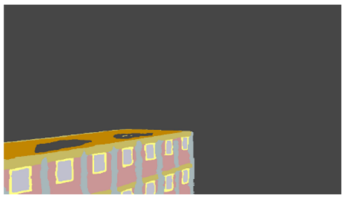
(e) IoU of spalling pixels of Modified U-Net++

Fig. 11 Change of IoU of each class with respect to different loss functions on local test dataset

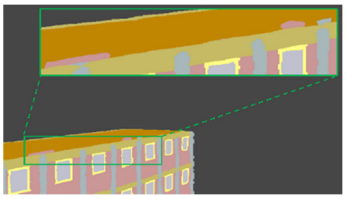
(a) Input image

(b) Ground truth mask

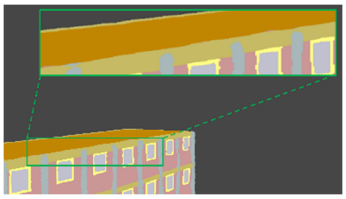
(c) Predicted mask using CE loss function



(d) Predicted mask using Focal loss function



(e) Predicted mask using Lovasz loss function



(f) Predicted mask using Lovasz-WCE loss function

Fig. 12 Comparison of predicted component masks from DeepLab-AASPP model when different loss functions are used. Pictures with green borders are the enlarged view of red bounding boxes

frame", "slab", "no damage" and "severe damage") improves a lot when switching from cross entropy based loss function to Lovasz-WCE loss function. This is because the IoU based loss function allows direct optimization towards the objective function and allocates heavierto underrepresented classes in the dataset. Therefore, IoU based functions perform better at class imbalance problems by design. However, the gradients of IoU based loss functions are usually not as smooth as the Cross Entropy based loss functions. To mitigate this issue, we use a combination of Lovasz and weighted Cross Entropy loss function $\mathcal{L}_{Lovasz-WCE}$ to better stabilize the training process.

Some representative prediction masks on component segmentation task are shown in Fig. 12. It can be observed in Fig. 12 (c) that the roof of the building is misclassified as background because "slab" is the minority class in component segmentation task. IoU of "slab" pixels further increases in Fig. 12(d) because Focal loss can dynamically scale the loss function to down-weight the background pixels and thus the network can focus on training "slab" pixels. Finally, when Lovasz loss functions are used in Figs. 12(d) and (f), the model can recover nearly all the "slab" pixels.

Table 6 mIoU/IoU score of different workflows in all 5 segmentation tasks

Workflow	Segmentation tasks				
	Comp.	DS	Crack	Spalling	Rebar
Patch based	0.9667	0.6890	0.5449	0.9375	0.5018
Full-scale image based	0.9850	0.7032	0.5113	0.9174	0.4571

4.2.3 Comparison of patch based and full-scale image based workflow

In this study, two different workflows, namely patch based workflow and full-scale image based workflow are used in different tasks. In full-scale image based workflow, the images are fed into the network and segmentation masks are predicted in an end-to-end fashion. Whereas in patch based workflow, a full-scale image is first cropped into overlapping patches. The image patches are then fed into the segmentation network to predict the masks. Finally, the predicted segmentation masks based on cropped patches are merged to form the final segmentation mask of the original image.

Table 6 lists out the performance difference in different tasks when different workflows are used. For component and damage state segmentation, the best performance is obtained from full-scale image based method, whereas patch based workflows give us a better performance on crack, spalling and rebar segmentation tasks. This is because cropping the image without caring about the nature of the data and tasks can have consequences on the quality of predictions. This is especially true when data points are highly correlated in image segmentation tasks. For example, in component and damage state segmentation tasks, the segmentation network might use global information like a column is usually perpendicular to a beam to predict the segmentation mask. Cropping the images without considering such spatial correlation can harm the prediction results. On the other hand, the distribution of damage areas usually does not have an obvious correlation. Therefore, cropping the images in crack, spalling and rebar segmentation tasks does not have visible negative effects on prediction results. As for why patch-based workflow works better than full-scale image-based workflow in detailed damage segmentation tasks, one explanation is that the non-damage patches are discarded during the cropping process, which implicitly balances the distribution between background and damage pixels.

4.2.4 Model complexity analysis

Including the channel-wise attention mechanism can effectively improve the mIoU score. However, this improvement comes with an increase in computational complexity. Let's consider one channel-wise attention module that outputs feature maps with C channels. Adding a channel-wise attention module with increase the computational complexity by C^2. Therefore, the increase in model complexity for the entire model is $\sum_{n=1}^{N} C_n^2$, where N is the total number of channel-wise attention modules used. For example, compared with vanilla DeepLabv3+, the proposed DeepLab-AASPP has 4.1×10^4 more parameters which is only 0.06% more than the vanilla model. For modified U-Net++, the total number of parameters is 1.4% more than the vanilla U-Net++ model. Since we only add a small fraction of parameters to the models, the increased computational time is negligible.

Table 7 IoU scores of damage pixels channel-wise attention modules placed at different locations inside the convolution block in Modified U-Net++

Tasks	Different configurations		
	Only at the beginning	Only at the end	Both beginning and the end
Crack	0.5427	0.5432	0.5449
Rebar	0.4998	0.5001	0.5018
Spalling	0.9366	0.9361	0.9375

4.2.5 Different configurations of channel-wise attention blocks

Although there are no significant improvements, it is observed that incorporating the channel-wise attention block in both the beginning and the end of the convolution block slightly enhances the IoU performance of U-Net++. As shown in Table 7, compared to the cases of having attention block solely in the beginning or the end of the convolution block, the proposed network architecture increases the IoU by approximately 0.1%.

Table 8 mIoU/IoU score using semi-supervised learning under different data partition protocols

Training method	Data partition protocols	DeepLab-ASPP		Modified U-Net++		
		Component (mIoU)	Damage state (mIoU)	Crack (IoU)	Rebar (IoU)	Spalling (IoU)
Supervised	1/8	0.9696	0.6115	0.5100	0.4034	0.9201
	1/4	0.9733	0.6546	0.5244	0.4553	0.9244
	1/2	0.9758	0.6801	0.5330	0.4867	0.9298
Semi-supervised	1/8	0.9749	0.6518	0.5204	0.4159	0.9237
	1/4	0.9783	0.6884	0.5329	0.4687	0.9271
	1/2	0.9790	0.7024	0.5408	0.4959	0.9318

4.3 Model performance under semi-supervised learning framework

During earthquake reconnaissance, the inspectors use UAVs or handheld cameras to collect thousands of images in a short time, but the labeling process usually takes a long time. Therefore, we take the scarcity of labeled data into consideration and proposed an SSL method to improve the performance of the segmentation network under the assumption that limited labeled images are available. A series of experiments are carried out under different data partition protocols and the results are shown in this section.

The improvement of the proposed SSL framework compared with the supervised learning baseline under different data partitions is given in Table 8. Table 8 shows that the SSL framework outperforms its supervised learning baseline consistently on all different tasks. For component segmentation, the SSL method can improve the mIoU score by 0.53%, 0.50%, and 0.32% under 1/8, 1/4, and 1/2 data partition respectively. Similarly for spalling segmentation, the mIoU score is improved 0.36%, 0.27%, and 0.20% compared with the supervised baseline. The mIoU improve 4.03%, 3.38% and 2.23% under 1/8, 1/4, and 1/2 partition protocols for damage state segmentation task. For crack segmentation task, the IoU of damage label improves 1.04%, 0.85%, and 0.78% under 1/8, 1/4, and 1/2 partition protocols. Finally, the IoU of Rebar improves 1.25%, 1.34%, and 0.92% respectively under 1/8, 1/4, and 1/2 partition protocols. The improvements in component segmentation and spalling segmentation are limited because the supervised baseline already quite performs well that it reaches the point of diminishing return (e.g., additional efforts yield very limited improvement) in terms of mIoU. Whereas the improvement brought by SSL is more obvious in tasks such as damage state, crack and rebar segmentation where the baseline IoU of supervised learning is relatively low. It should be noted that the improvement is more significant when less labeled data is available, for example, the improvement under the 1/8 partition protocol is higher than the 1/2 partition protocol. It is expected that the lack of ground truth labels in supervised learning can make the network overfits to the small training data and performs badly on test images, whereas the SSL can make use of the extra unlabeled data to train the network. The supervised loss $\mathcal{L}_s$ combined with unsupervised loss $\mathcal{L}_u$ encourage the consistent prediction between unlabeled images while making correct prediction on the labeled data.

From Table 8, it can be observed that the proposed SSL method could achieve similar accuracy using much fewer labeled data than the supervised learning baseline. For example, SSL only uses 796 labeled images (1/4 of the whole labeled set) to achieve the mIoU of 0.6884 in damage state segmentation task, while the supervised learning baseline needs 1,592 labeled images (1/2 of the whole labeled set) to achieve a similar mIoU. The bold numbers are the cases where the performance improved the most when comparing semi-supervised learning to supervised learning.

Despite the improvement in the model performances, it would be interesting to look into the training progress and analysis how the proposed SSL approach works. In the early stage of the training, there must be cases where pixels are wrongly predicted in both networks $f(\theta_1)$ and $f(\theta_2)$. As mentioned in Section 3.5, cross supervision is used in the unsupervised part of the loss function $\mathcal{L}_{semi}$, which means the prediction $\hat{\boldsymbol{y}}_1$ from $f(\theta_1)$ is used to supervise the training of $f(\theta_2)$, and the prediction $\hat{\boldsymbol{y}}_2$ from $f(\theta_2)$ is used to supervise the training of $f(\theta_1)$. Let's consider the case where $\hat{\boldsymbol{y}}_1$ is used to train network $f(\theta_2)$, and we can further split it into the following four cases: (1) Both $f(\theta_1)$ and $f(\theta_2)$ predict the correct label; (2) $f(\theta_1)$ predicts the correct label and $f(\theta_2)$ predicts the wrong label; (3) $f(\theta_1)$ predicts the wrong label and $f(\theta_2)$ predicts the correct label; (4) Both $f(\theta_1)$ and $f(\theta_2)$ predict the wrong label. In (1) and (2), $\hat{\boldsymbol{y}}_1$ is correct, so the prediction will guide the training towards the right direction. In (3), the supervision signal from $\hat{\boldsymbol{y}}_1$ is wrong and it can lead the training towards the wrong direction. In (4), both predictions are wrong, but the predictions are penalized unless both networks predict the same wrong label at the same time. If both networks predict the same wrong label, then it could harm the training process. Compared with multiclass segmentation, it is more likely for both networks to predict the same wrong label in the binary segmentation task because there are fewer classes for the networks to choose from. This is why the improvement from crack, rebar and spalling segmentation task is less than the damage state segmentation task. Both (3) and (4) could have a negative influence on the training but the supervised loss $\mathcal{L}_s$ in $\mathcal{L}_{semi}$ could mitigate this issue. To sum up, our hypothesis is that the advantages outweigh the disadvantage in the above-mentioned cases, which brings improvement to the IoU scores when lacking the ground truth label. The SSL algorithm could be further improved if a threshold selection can be introduced to remove the bad supervision signal from low-confidence predictions during cross supervision training.

5. Conclusions

In this study, two modified architectures based on channel-wise attention mechanism are proposed to perform segmentation tasks on virtual post-earthquake buildings. In addition, an SSL framework is proposed to improve the performance of the segmentation network under the assumption that limited labeled data is available. To tackle the data imbalance problem, different loss functions are studied and compared. Experiments are conducted based on 500 test images and the major findings are as follows:

- A channel-wise attention module is proposed and embedded in the DeepLabv3+ and U-Net++ to enhance the segmentation performance. With additional channel-wise attention modules and structural modifications, DeepLab-AASPP achieves the best performance on component segmentation and damage state segmentation tasks with mIoU scores of 0.9850 and 0.7032. For crack, spalling and rebar segmentation tasks, modified U-Net++ obtains the best performance with mIoU scores of 0.5449,

0.9375, and 0.5018 on these three separate segmentation tasks.

- Different loss functions are implemented and compared. Lovasz-WCE loss function performs the best among all the loss functions in different tasks. Compared with Cross Entropy loss function, Lovasz-WCE loss function increase the mIoU by 3.16%, 4.8%, 11.4%, 14.1%, and 7.1% respectively on component, damage state, crack, spalling and rebar segmentation tasks.
- An SSL method based on consistency regularization and cross supervision is proposed. The proposed method can make use of unlabeled images to further improve the segmentation performance. An improvement of up to 4.03% is observed in damage state segmentation task, and different levels of improvement are observed in other tasks.
- The proposed SSL method needs much fewer images to achieve similar performance compared with the supervised learning method. Therefore, the proposed SSL methods can greatly reduce the manual labor of labeling, making it more efficient to train and deploy an inspection network during quick post-earthquake reconnaissance.
- It is observed that the patch based workflow can benefit the crack, spalling and rebar segmentation tasks. However, for component and damage state segmentation tasks, preserving the spatial correlation is critical, and therefore cropping the image imposes negative effects on the model performances.
- For future studies, the SSL method could be further improved if a threshold selection can be introduced to remove the bad cross supervision signal from low-confidence predictions during cross supervision training.

Acknowledgments

The authors would like to thank the organization of the IC-SHM 2021: ANCRiSST, University of Illinois at Urbana-Champaign, Harbin Institute of Technology, Zhejiang University, and University of Houston for providing the valuable data used in this study.

References

Abdel-Qader, I., Abudayyeh, O. and Kelly, M.E. (2003), "Analysis of edge-detection techniques for crack identification in bridges", *J. Computer Civil Eng.*, **17**(4), 255-263. https://doi.org/10.1061/(ASCE)0887-3801(2003)17:4(255)

Bao, Y. and Li, H. (2021), "Machine learning paradigm for structural health monitoring", *Struct. Health Monitor.*, **20**(4), 1353-1372. https://doi.org/10.1177/1475921720972416

Bao, Y., Chen, Z., Wei, S., Xu, Y., Tang, Z. and Li, H. (2019), "The state of the art of data science and engineering in structural health monitoring", *Engineering*, **5**(2), 234-242. https://doi.org/10.1016/j.eng.2018.11.027

Beckman, G.H., Polyzois, D. and Cha, Y.J. (2019), "Deep learning-based automatic volumetric damage quantification using depth camera", *Automat. Constr.*, **99**, 114-124. https://doi.org/10.1016/j.autcon.2018.12.006

Berman, M., Triki, A.R. and Blaschko, M.B. (2018), "The lovász-softmax loss: A tractable surrogate for the optimization of the intersection-over-union measure in neural networks", *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4413-4421.

Bermanmaxim. Bermanmaxim/LovaszSoftmax: Code for the lovász-softmax loss (CVPR 2018), Retrieved January 28, 2022. https://github.com/bermanmaxim/LovaszSoftmax

Cha, Y.J., Choi, W. and Büyüköztürk, O. (2017), "Deep learning-based crack damage detection using convolutional neural networks", *Comput.-Aided Civil Infrastr. Eng.*, **32**(5), 361-378. https://doi.org/10.1111/mice.12263

Cha, Y.J., Choi, W., Suh, G., Mahmoudkhani, S. and Büyüköztürk, O. (2018), "Autonomous structural visual inspection using region-based deep learning for detecting multiple damage types", *Comput.-Aided Civil Infrastr. Eng.*, **33**(9), 731-747. https://doi.org/10.1111/mice.12334

Chen, F.C. and Jahanshahi, M.R. (2017), "NB-CNN: Deep learning-based crack detection using convolutional neural network and Naïve Bayes data fusion", *IEEE Transact. Indust. Electron.*, **65**(5), 4392-4400. https://doi.org/10.1109/TIE.2017.2764844

Chen, L.C., Zhu, Y., Papandreou, G., Schroff, F. and Adam, H. (2018), "Encoder-decoder with atrous separable convolution for semantic image segmentation", *Proceedings of the European Conference on Computer Vision* (*ECCV*), pp. 801-818.

Chen, X., Yuan, Y., Zeng, G. and Wang, J. (2021), "Semi-supervised semantic segmentation with cross pseudo supervision", *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2613-2622.

Cheng, H.D., Chen, J.R., Glazier, C. and Hu, Y.G. (1999), "Novel approach to pavement cracking detection based on fuzzy set theory", *J. Computer Civil Eng.*, **13**(4), 270-280. https://doi.org/10.1061/(ASCE)0887-3801(1999)13:4(270)

Cheng, H.D., Shi, X.J. and Glazier, C. (2003), "Real-time image thresholding based on sample space reduction and interpolation approach", *J. Computer Civil Eng.*, **17**(4), 264-272. https://doi.org/10.1061/(ASCE)0887-3801(2003)17:4(264)

Choi, W. and Cha, Y.J. (2019), "SDDNet: Real-time crack segmentation", *IEEE Transact. Industr. Electron.*, **67**(9), 8016-8025. https://doi.org/10.1109/TIE.2019.2945265

Deng, J., Dong, W., Socher, R., Li, L.J., Li, K. and Fei-Fei, L. (2009), "Imagenet: A large-scale hierarchical image database", *Proceedings of 2009 IEEE Conference on Computer Vision and Pattern Recognition*, Miami, FL, USA, June, pp. 248-255. https://doi.org/10.1109/CVPR.2009.5206848

Dong, C., Li, L., Yan, J., Zhang, Z., Pan, H. and Catbas, F.N. (2021), "Pixel-level fatigue crack segmentation in large-scale images of steel structures using an encoder–decoder network", *Sensors*, **21**(12), 4135. https://doi.org/10.3390/s21124135

Dung, C.V. (2019), "Autonomous concrete crack detection using deep fully convolutional neural network", *Automat. Constr.*, **99**, 52-58. https://doi.org/10.1016/j.autcon.2018.11.028

Fujita, Y. and Hamamoto, Y. (2011), "A robust automatic crack detection method from noisy concrete surfaces", *Mach. Vis. Applicat.*, **22**(2), 245-254. https://doi.org/10.1007/s00138-009-0244-5

Gao, Y. and Mosalam, K.M. (2018), "Deep transfer learning for image-based structural damage recognition", *Comput.-Aided Civil Infrastr. Eng.*, **33**(9), 748-768. https://doi.org/10.1111/mice.12363

Ghosh Mondal, T., Jahanshahi, M.R., Wu, R.T. and Wu, Z.Y. (2020), "Deep learning-based multi-class damage detection for autonomous post-disaster reconnaissance", *Struct. Control Health Monitor.*, **27**(4), e2507. https://doi.org/10.1002/stc.2507

He, K., Zhang, X., Ren, S. and Sun, J. (2015), "Spatial pyramid

pooling in deep convolutional networks for visual recognition", *IEEE Transact. Pattern Anal. Mach. Intell.*, **37**(9), 1904-1916. https://doi.org/10.1109/TPAMI.2015.2389824

He, K., Zhang, X., Ren, S. and Sun, J. (2016), "Deep residual learning for image recognition", *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770-778.

Hoskere, V., Narazaki, Y., Hoang, T.A. and Spencer Jr, B.F. (2020), "MaDnet: multi-task semantic segmentation of multiple types of structural materials and damage in images of civil infrastructure", *J. Civil Struct. Health Monitor.*, **10**, 757-773. https://doi.org/10.1007/s13349-020-00409-0

Hoskere, V., Narazaki, Y. and Spencer, B.F. (2022), "Physics-based graphics models in 3D synthetic environments as autonomous vision-based inspection testbeds", *Sensors*, **22**(2), 532. https://doi.org/10.3390/s22020532

Hu, J., Shen, L. and Sun, G. (2018), "Squeeze-and-excitation networks", *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7132-7141.

Ji, A., Xue, X., Wang, Y., Luo, X. and Xue, W. (2020), "An integrated approach to automatic pixel-level crack detection and quantification of asphalt pavement", *Automat. Constr.*, **114**, 103176. https://doi.org/10.1016/j.autcon.2020.103176

Ke, Z., Qiu, D., Li, K., Yan, Q. and Lau, R.W. (2020), "Guided collaborative training for pixel-wise semi-supervised learning", *Proceedings of European Conference on Computer Vision*, pp. 429-445. https://doi.org/10.1007/978-3-030-58601-0_26

Lau, S.L., Chong, E.K., Yang, X. and Wang, X. (2020), "Automated pavement crack segmentation using u-net-based convolutional neural network", *IEEE Access*, **8**, 114892-114899. https://doi.org/10.1109/ACCESS.2020.3003638

Lin, T.Y., Goyal, P., Girshick, R., He, K. and Dollár, P. (2017), "Focal loss for dense object detection", *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2980-2988.

Liu, Y., Yao, J., Lu, X., Xie, R. and Li, L. (2019), "DeepCrack: A deep hierarchical feature learning architecture for crack segmentation", *Neurocomputing*, **338**, 139-153. https://doi.org/10.1016/j.neucom.2019.01.036

Liu, J., Yang, X., Lau, S., Wang, X., Luo, S., Lee, V.C.S. and Ding, L. (2020), "Automated pavement crack detection and segmentation based on two-step convolutional neural network", *Comput.-Aided Civil Infrastr. Eng.*, **35**(11), 1291-1305. https://doi.org/10.1111/mice.12622

Long, J., Shelhamer, E. and Darrell, T. (2015), "Fully convolutional networks for semantic segmentation", *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Boston, MA, USA, June, pp. 3431-3440.

Maeda, H., Sekimoto, Y., Seto, T., Kashiyama, T. and Omata, H. (2018), "Road damage detection and classification using deep neural networks with smartphone images", *Comput.-Aided Civil Infrastr. Eng.*, **33**(12), 1127-1141. https://doi.org/10.1111/mice.12387

Narazaki, Y., Hoskere, V., Eick, B.A., Smith, M.D. and Spencer, B.F. (2019), "Vision-based dense displacement and strain estimation of miter gates with the performance evaluation using physics-based graphics models", *Smart Struct. Syst., Int. J.*, **24**(6), 709-721. https://doi.org/10.12989/sss.2019.24.6.709

Narazaki, Y., Hoskere, V., Yoshida, K., Spencer, B.F. and Fujino, Y. (2021), "Synthetic environments for vision-based structural condition assessment of Japanese high-speed railway viaducts", *Mech. Syst. Signal Process.*, **160**, 107850. https://doi.org/10.1016/j.ymssp.2021.107850

Ouali, Y., Hudelot, C. and Tami, M. (2020), "Semi-supervised semantic segmentation with cross-consistency training", *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12674-12684.

Paszke, A., Chaurasia, A., Kim, S. and Culurciello, E. (2016), "ENet: A deep neural network architecture for real-time semantic segmentation", arXiv preprint arXiv:1606.02147. https://doi.org/10.48550/arXiv.1606.02147

Ronneberger, O., Fischer, P. and Brox, T. (2015), "U-net: Convolutional networks for biomedical image segmentation", *Proceedings of International Conference on Medical Image Computing and Computer-Assisted Intervention*, October, pp. 234-241. https://doi.org/10.1007/978-3-319-24574-4_28

Sajedi, S.O. and Liang, X. (2021), "Uncertainty-assisted deep vision structural health monitoring", *Comput.-Aided Civil Infrastr. Eng.*, **36**(2), 126-142. https://doi.org/10.1111/mice.12580

Spencer Jr, B.F., Hoskere, V. and Narazaki, Y. (2019), "Advances in computer vision-based civil infrastructure inspection and monitoring", *Engineering*, **5**(2), 199-222. https://doi.org/10.1016/j.eng.2018.11.030

Tang, W., Wu, R.-T. and Jahanshahi, M.R. (2022), "Crack segmentation in high-resolution images using cascaded deep convolutional neural networks and Bayesian data fusion", *Smart Struct. Syst., Int. J.*, **29**(1), 221-235. https://doi.org/10.12989/sss.2022.29.1.221

Xue, Y. and Li, Y. (2018), "A fast detection method via region-based fully convolutional neural networks for shield tunnel lining defects", *Comput.-Aided Civil Infrastr. Eng.*, **33**(8), 638-654. https://doi.org/10.1111/mice.12367

Yamaguchi, T. and Hashimoto, S. (2010), "Fast method for crack detection surface concrete large-size images using percolation-based image processing", *Mach. Vis. Appl.*, **21**, 797-809.

Yang, Q. and Ji, X. (2021), "Automatic pixel-level crack detection for civil infrastructure using UNet++ and deep transfer learning", *IEEE Sensors J.*, **21**(17), 19165-19175. https://doi.org/10.1109/JSEN.2021.3089718

Yang, X., Li, H., Yu, Y., Luo, X., Huang, T. and Yang, X. (2018), "Automatic pixel-level crack detection and measurement using fully convolutional network", *Comput.-Aided Civil Infrastr. Eng.*, **33**(12), 1090-1109. https://doi.org/10.1111/mice.12412

Zhang, X., Rajan, D. and Story, B. (2019), "Concrete crack detection using context-aware deep semantic segmentation network", *Comput.-Aided Civil Infrastr. Eng.*, **34**(11), 951-971. https://doi.org/10.1111/mice.12477

Zhou, Z., Siddiquee, M.M.R., Tajbakhsh, N. and Liang, J. (2018), "Unet++: A nested u-net architecture for medical image segmentation", In: *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support*, pp. 3-11.

Zou, Y., Zhang, Z., Zhang, H., Li, C.L., Bian, X., Huang, J.B. and Pfister, T. (2020), "Pseudoseg: Designing pseudo labels for semantic segmentation", arXiv preprint arXiv:2010.09713. https://doi.org/10.48550/arXiv.2010.09713