Smart Structures and Systems, *Vol. 31*, *No. 4* (*2023*) *351-363*
https://doi.org/10.12989/sss.2023.31.4.351

Twin models for high-resolution visual inspections

Seyedomid Sajedi [a‡], Kareem A. Eltouny [b‡] and Xiao Liang*

Department of Civil, Structural and Environmental Engineering,
University at Buffalo, the State University of New York, Buffalo, New York, 14260, USA

(*Received August 30, 2022, Revised November 20, 2022, Accepted February 2, 2023*)

Abstract. Visual structural inspections are an inseparable part of post-earthquake damage assessments. With unmanned aerial vehicles (UAVs) establishing a new frontier in visual inspections, there are major computational challenges in processing the collected massive amounts of high-resolution visual data. We propose twin deep learning models that can provide accurate high-resolution structural components and damage segmentation masks efficiently. The traditional approach to cope with high memory computational demands is to either uniformly downsample the raw images at the price of losing fine local details or cropping smaller parts of the images leading to a loss of global contextual information. Therefore, our twin models comprising Trainable Resizing for high-resolution Segmentation Network (TRS-Net) and DmgFormer approaches the global and local semantics from different perspectives. TRS-Net is a compound, high-resolution segmentation architecture equipped with learnable downsampler and upsampler modules to minimize information loss for optimal performance and efficiency. DmgFormer utilizes a transformer backbone and a convolutional decoder head with skip connections on a grid of crops aiming for high precision learning without downsizing. An augmented inference technique is used to boost performance further and reduce the possible loss of context due to grid cropping. Comprehensive experiments have been performed on the 3D physics-based graphics models (PBGMs) synthetic environments in the QuakeCity dataset. The proposed framework is evaluated using several metrics on three segmentation tasks: component type, component damage state, and global damage (crack, rebar, spalling). The models were developed as part of the 2nd International Competition for Structural Health Monitoring.

Keywords: computer vision; crack detection; damage detection; deep learning; IC-SHM; semantic segmentation; visual inspections

1. Introduction

There is no doubt that recent breakthroughs in deep learning, accompanied by advances in computing and sensing technology, have significantly impacted Structural Health Monitoring (SHM) research, especially in autonomous visual inspections. Camera-equipped Unmanned Aerial Vehicles (UAVs) are becoming more available in the inspection industry and assist with collecting high-resolution data from areas that are difficult or unsafe to access by human inspectors (Dorafshan *et al*. 2018, Liu *et al*. 2020, Narazaki *et al*. 2022). The collected footage is often manually examined by human inspectors for structural defects which could be suitable in a small-scale inspection task. However, this can cause a recovery bottleneck if applied to major urban areas struck by a natural catastrophe. Therefore, there has been a research focus on developing autonomous vision-based structural health monitoring systems. Conventional cracks and spalling detection systems usually involve morphological operations, edge detection algorithms, filtering, binarization, and other image processing techniques (Abdel-Qader *et al*. 2003, Jahanshahi and Masri 2012, Li *et al*. 2018, Talab *et al*. 2016). In the past decade, however, machine learning algorithms have been favored by researchers for visual damage detection given the recent advances in artificial intelligence (Munawar *et al*. 2021).

Deep learning technology allows to post-process massive amounts of visual data obtained from drone footage in the presence of reliable models. Generally, there are two main directions in developing vision-based SHM: component detection and damage detection. For detecting building components, structural or non-structural, multiple methods have been proposed in the past using various types including object detection (Agyemang *et al*. 2021, Hou *et al*. 2020, Liang 2019), and semantic segmentation (Narazaki *et al*. 2020, Sajedi and Liang 2021). The acquired predictions can accelerate the inspection process by providing guidance to both inspectors and UAVs. Considerable efforts were made in developing machine learning algorithms for detecting various types of structural damage. Different deep learning models have been proposed for concrete cracks including U-Net (Liu *et al*. 2019), YOLOv2 (Teng *et al*. 2021), deep learning models with Bayesian inference (Sajedi and Liang 2021), cascaded deep convolutional neural networks (CNNs) (Tang *et al*. 2022), and stacked convolutional autoencoders (Zheng *et al*. 2022). Pavement crack detection has also been

*Corresponding author, Ph.D., Assistant Professor,
E-mail: liangx@buffalo.edu
[a] Ph.D., AI Engineer
[b] Ph.D. Candidate
‡ Equal Contribution

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

investigated using deep CNNs with transfer learning (Zhang *et al*. 2018), the CNN-based CrackNet (Zhang *et al*. 2017), and SegNet-like architectures (Sajedi and Liang 2019). Detecting the defects of rail surfaces (Wu *et al*. 2022) and building façades (Guo *et al*. 2021) were also explored. In addition, a number of models were developed for multi-task damage detection including fatigue cracks, spalling, corrosion, and other visual inspection tasks (Hoskere *et al*. 2017, 2020, Zhou *et al*. 2022).

A common approach in dealing with the SHM vision problem is to adopt off-the-shelf deep learning algorithms and use them with techniques such as transfer learning or training from scratch. While this approach can be a working solution in many tasks, there are delicacies in structural inspections that need to be considered. Autonomous visual inspections need to have high precision, accuracy, and computational efficiency.

Limited resources often dictate downsizing images drastically for different tasks such as semantic segmentation. However, critical information might be lost, especially in the presence of structural damage. We aim to develop a unified visual inspection framework that can strike a balance between prediction quality and computational efficiency for various tasks, including semantic segmentation of building components, their damage states, cracks, spalling, and exposed rebars. Given the different nature of these tasks, we propose a twin-model framework that approaches these tasks from two perspectives. TRS-Net is developed for component detection and damage severity segmentation, which highly rely on global context information. DmgFormer is dedicated to crack, rebar, and spalling segmentation, tasks that are highly sensitive to the loss of resolution. The proposed twin models are customized based on the latest advances in deep learning computer vision research on super-resolution, image segmentation, and transformer architectures. We further evaluate our framework on the QuakeCity dataset (Hoskere *et al*. 2022).

The remainder of this paper is organized as follows. The following section provide insights on TRS-Net and DmgFormer architectures and network design optimization. After discussing the theory and notions behind each model, a section is dedicated to describing the case study, the implementation details of the twin models, and the results to document the testing evaluation metrics. The last part of this paper concludes the benefits and limitations of each model for different vision tasks.

2. Architecture design

The majority of the deep learning semantic segmentation models are optimized for low to medium-resolution images dataset. When dealing with high-resolution images, it is common to uniformly downsample high-resolution images, as a preprocessing step, to be fed into segmentation models for training and inference (Katharopoulos and Fleuret 2019). The predicted masks can be either used in their downsized versions or upsampled to the original size. Downsampling/upsampling operations are typically performed uniformly or by using an interpolation

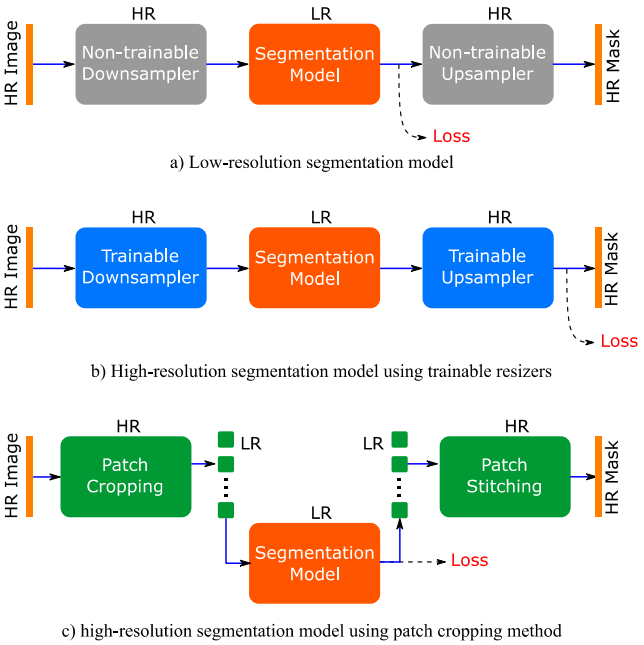


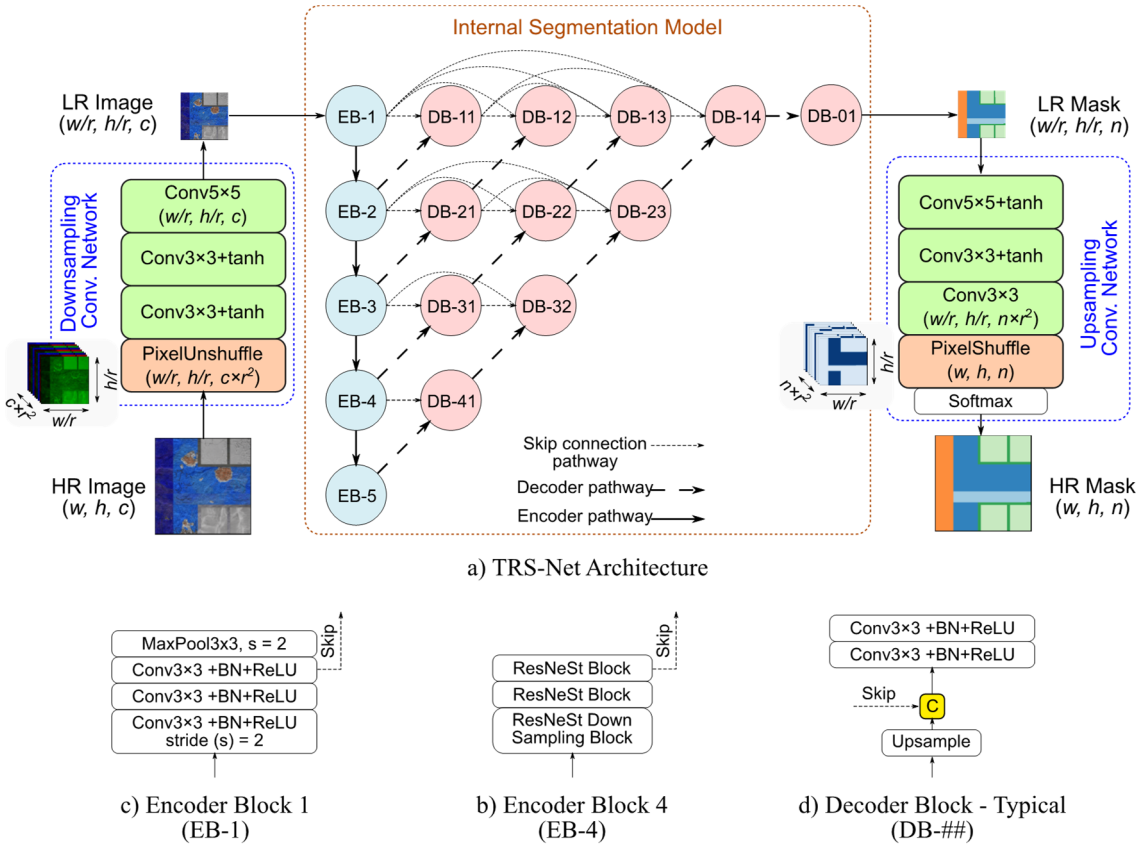
Fig. 1 High-resolution segmentation approaches (HR: high resolution; LR: low resolution)

method (e.g., nearest neighbor). Training is then performed on low-resolution images and masks (Fig. 1(a)). However, these operations result in poor segmentation performance near the object edges as a result of giving equal importance to all pixels in the image. This problem is even more severe for damage segmentation of rebars, spalls, and cracks. Moreover, if the segmentation networks (e.g., U-Net) are to operate directly on high-resolution images, the model is met by GPU-memory constraints and could become computationally inefficient.

To tackle these challenges, we aim to provide high-fidelity segmentation masks with precise edges for building components and their damage states. At the same time, we also aim to maintain the efficiency of low-resolution segmentation models and satisfy GPU-memory constraints. We explore two approaches and implement and use each one of them for different visual inspection segmentation objectives. The first method is using downsamplers and upsamplers of trainable parameters inspired by advances in super-resolution models (Fig. 1(b)). The full model can then be trained end-to-end based on high-resolution images and masks. This approach is suitable for understanding global semantics as the structure of the image and its components remain intact during training. The second approach is relying on patch cropping the image in a grid-like fashion to preserve the local and fine details such as fine cracks and exposed rebars, especially if the images were captured from distance (Fig. 1(c)). As observed, the two models complement each other considering the different tasks in visual inspection, and together, they create our twin-model high-resolution visual inspection framework.

2.1 TRS-Net for components and damage state segmentations

In the first part of the twin-model framework, we



a) TRS-Net Architecture

c) Encoder Block 1 (EB-1)

b) Encoder Block 4 (EB-4)

d) Decoder Block - Typical (DB-##)

Fig. 2 TRS-Net architecture. (HR: high-resolution, LR: low-resolution, Conv: 2D convolution, BN: batch normalization, ReLU: rectified linear unit, EB-*i* encoder block, DB-*ij*: decoder block, *c*: RGB channels, *n*: classes, *r*: scaling factor)

propose trainable resizing for high-resolution image segmentation. It is a compound segmentation model with two-layered encoder-decoder networks (Fig. 2). The network is ideal for components and damage state segmentation as well as other tasks that benefit from prioritizing global contextual information. The outer encoder-decoder network includes efficient trainable downsampler and upsampler modules. Their primary goals are to learn to focus on the important regions and pixels during the downsizing process and also to reconstruct the high-resolution masks. TRS-N*et al*so has an internal encoder-decoder which is a U-Net-style semantic segmentation model trained to predict pixel-wise model end-to-end. The following four subsections describe the outer encoder-decoder modules (trainable resizers), the backbone of the segmentation model, the internal segmentation model, and finally a brief discussion on the memory efficiency of the network.

2.1.1 The outer encoder-decoder modules: Learning to resize images and masks

The outer part of the model is inspired by the advances made in image super-resolution networks; in particular, the efficient sub-pixel convolution network (Shi *et al*. 2016). The main ingredients in these trainable resizers are the pixel shuffle and unshuffle operations. Unlike pooling and interpolation-based resizing operations, pixel shuffle/unshuffle does not alter the values of the feature maps when resizing the feature maps width and height but only rearranges the pixels in the channel dimensions.

The proposed upsampler and downsampler are almost a mirror of one another and details regarding their composition are in Table 1. The upsampler, called thereafter upsampling convolutional network (UCN), is composed of

Table 1 DCN and UCN specification

	DCN (Downsampler)			UCN (Upsampler)		
Layer #	Operator	Kernel	# Channels	Operator	Kernel	# Channels
1	PixelUnshuffle	4×4	48	Conv.	5×5	64
2	Conv.	3×3	32	Conv.	3×3	32
3	Conv.	3×3	64	Conv.	3×3	$n\times4^2$
4	Conv.	5×5	3	PixelShuffle	4×4	n

*n is the number of classes/masks

Table 2 TRS-Net Internal encoder-decoder specifications

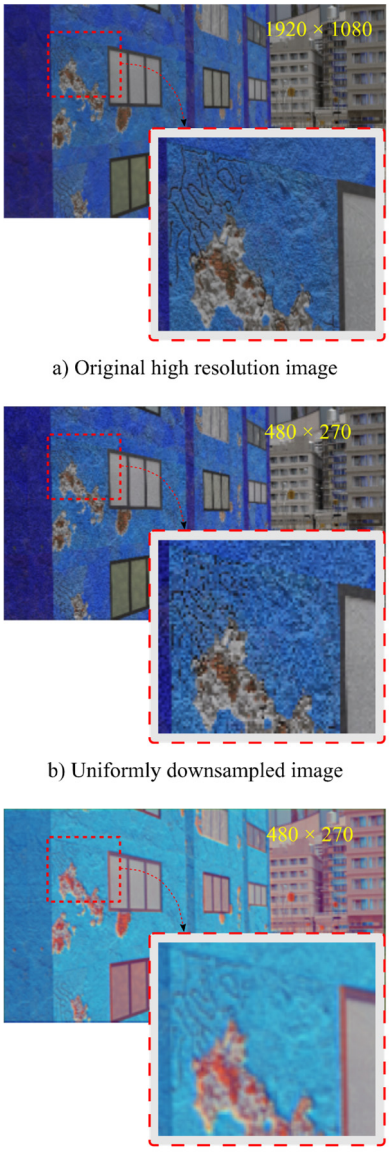
	Encoder				Decoder # Kernels		
Stage	Operator	# Channels	Cardinals width	# Layers	Last DB	Skip connection DB	# Spatial dim.
Head	-	-	-	1	*n* layer	-	480×288
0		-	-	1	16	-	480×288
1	Conv 3×3	64	-	3	32	64	240×144
2	ResNeSt	256	40	3	64	256	120×72
3	ResNeSt	512	160	4	128	512	60×36
4	ResNeSt	1024	320	6	256	-	30×18
5	ResNeSt	2048	640	3	-	-	15×9

*n is the number of classes/masks

three single-stride convolution layers followed by a pixel shuffle layer.

The pixel shuffle layer aggregates the low-resolution (w/r, h/r, $n\times r^2$) feature maps to form r-times upscaled masks (w,h,n). r is the scale factor, taken as 4 in this model, and n represents the number of classes. A final activation, such as SoftMax, is attached after the pixel shuffle layer depending on the classification type. The downsampling convolutional network (DCN), on the contrary, starts with a pixel unshuffled layer, a reverse operation to pixel shuffle, ensuring that no pixels are lost in the downsizing process but are rather arranged in the channels' dimension. Pixel unshuffled transforms a high-resolution image (w, h, c) to a r-times downsampled one (w/r, h/r, $c\times r^2$) but with r^2-times more channels. This layer is again followed by three convolutions, with the last one providing three feature maps for the internal segmentation model.

To demonstrate the benefits of using the trainable resizing modules, we show a comparison between the commonly used non-trainable uniform downsampling process and our proposed DCN on an example image from the QuakeCity dataset (Fig. 3). Compared to the original high-resolution image in Fig. 3(a), there is a significant loss of valuable information when performing a quarter-scale uniform downsampling. The cracks are harder to detect in the zoomed-in region in Fig. 3(b), and the window edges are significantly distorted. This makes it very difficult for the internal segmentation model to provide high-quality component detection and damage-state predictions. Using DCN (Fig. 3(c)), the downsampler can learn the importance of these pixels and retain them in a meaningful way in low-resolution. In addition, it can also emphasize some of the important edges in the downsized image, which are even harder to detect in the original image. A good example would be the upper beam edge in the focused region in Fig. 3. This is also seen in the cracks as they appear thicker in our DCN module than in the original image. This characteristic makes DCN equivalent to a non-uniform sampler that has a warped sampling grid with dense sampling near relevant pixels compared to others.



a) Original high resolution image

b) Uniformly downsampled image

c) Our downsampler module

Fig. 3 An example comparison between different downsampling techniques. Our trainable downsampler (c) can retain most of the fine details, such as cracks, as well as edges that are usually distorted or lost when using a uniform downsampling operation

2.1.2 Split attention blocks (ResNeSt)

One of the main building blocks for the internal segmentation model of TRS-Net is ResNeSt (Fig. 4).

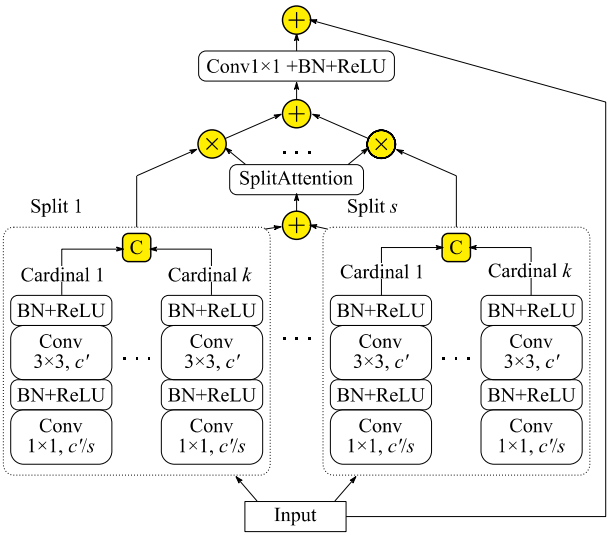


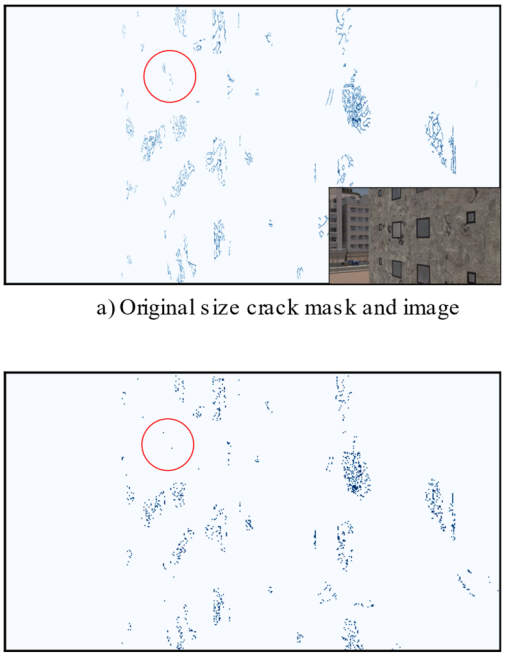
Fig. 4 Split Attention (ResNeSt) Block – reproduced from (Zhang *et al*. 2020) (Conv: 2D convolution, BN: batch normalization, ReLU: rectified linear unit, *c*': cardinal width, *k*: cardinality, *s*: radix)

ResNeSt is a family of state-of-the-art image classifier networks that has similar architectures to ResNet (He *et al*. 2016) but with different building blocks. ResNeSt was designed with efficiency in mind and had a similar inference speed as EfficientNet (Tan and Le 2019), with slightly improved classification performance. Based on the concept of feature map groups (also known as cardinality), ResNeSt splits the feature maps of the first two convolutional layers into $s \times k$ groups where s and k represent the number of splits (radix) and cardinals (cardinality), respectively. They are organized so that every k cardinals are grouped into a single split. The feature maps of all cardinals in a split are concatenated, and the s splits feature maps are aggregated through summation. Based on the Squeeze and Excitation (SE) layer from SE-Net (Hu *et al*. 2018), the feature maps pass through a split-attention layer which calibrates the splits feature maps. The primary purpose of the split-attention module is to capture the global context of the input by learning the channels' interdependencies. More details on the split-attention module can be found in Zhang *et al*. (2022). The split feature map groups are multiplied by the channel-wise split-attention factors and then aggregated through summation. Finally, and after the 1×1 convolution, a skip connection from the input combines the input with the output by addition.

2.1.3 Internal segmentation model

The internal encoder-decoder module is a low-resolution segmentation model based on a U-Net++ decoder, implemented by Yakubovskiy (2020), with a variant of the ResNeSt50d encoder. The network includes multiple pathways that can be categorized into encoder, decoder, and skip-connection pathways. It is also organized into five encoder blocks (EB-*i*) and 11 decoder blocks (DB-*ij*), where *i* indicates the encoder stage and *j* indexes the decoder's dense block layer. Details for each block are in Table 2. The main difference between the used U-Net++ architecture and the original U-Net is the complex dense convolutional blocks added to the skip-connection from the encoder to the decoder layers. Furthermore, additional decoder pathways are added, connecting the dense blocks with the upper encoder/decoder stage. The added convolutional layers and dense connections along the skip connections pathway can bridge the semantic gap between the encoder and decoder and improve gradient flow (Zhou *et al*. 2018). The encoder pathway passes through five stages of encoder blocks EB-1 to EB-5. The spatial dimensions of the segmentation model input must be a multiple of 32 due to having five stages of pooling in the model. Thus, zero-padding is applied to the DCN output spatial dimensions and removed from the UCN input to be compatible.

In this network, we adopt the ResNeSt50d_4s2x40d variant, implemented by Wightman (2019), as the backbone of the segmentation model. It has a cardinality of two, four splits, and a cardinality base width of 40 channels. This variant offers improvement compared to the original ResNeSt50d with a comparable number of parameters. The general architecture of the encoder stages follows the ResNetD architecture proposed by He *et al*. (2019). Aside from EB-0, each encoder is built from several consecutive ResNeSt and ResNeSt downsampling blocks. The ResNeSt downsampling block, which is used as the first layer in EB-2 to EB-5, adds the global average pooling layer before the final convolution layer as well as in the input skip connection. The decoder blocks receive their input from the skip-connection pathway (except for DB-01) and the decoder pathway. In each block, the inputs are concatenated after upsampling the decoder pathway input (using the nearest neighbor) and are then passed to two convolutional layers. The decoder head includes a single convolutional



a) Original size crack mask and image

b) Resampled crack mask

Fig. 5 Loss of damage information after downsizing the image to 25% of its dimensions and subsequently upsampling to the original resolution

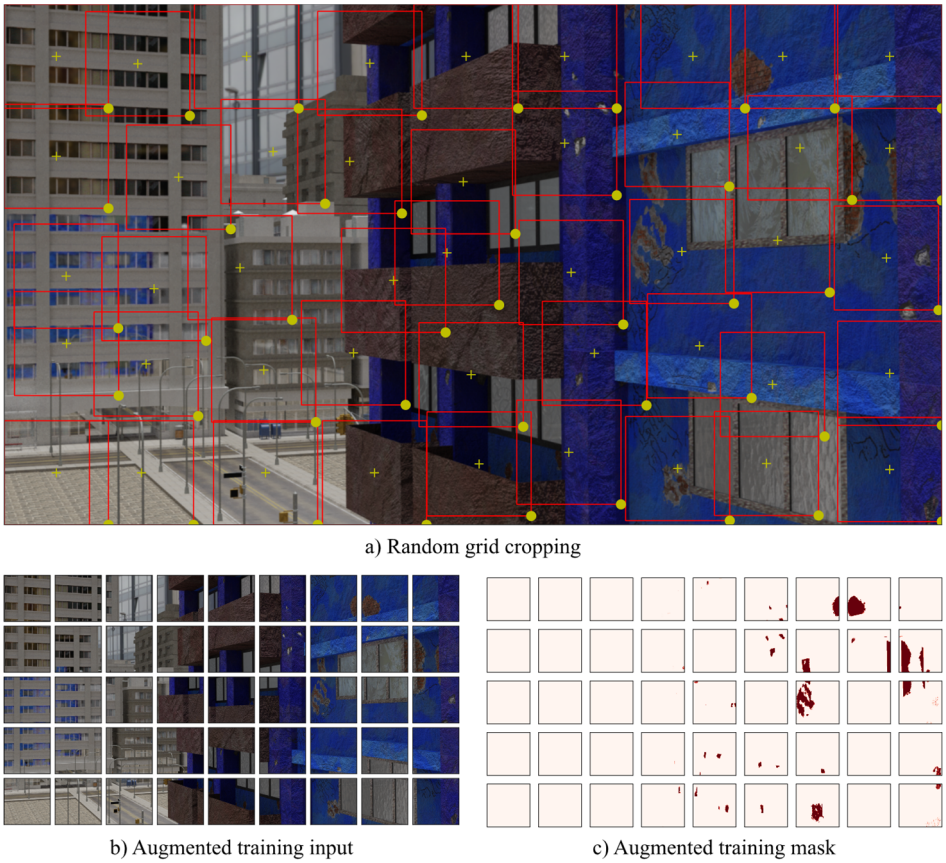


Fig. 6 Demonstration of Random grid cropping to augment the training dataset. Each crop's centroid and bottom right corner are depicted with + and o markers

layer that provides a set of feature maps corresponding to the number of classes. All convolutional layers in the internal segmentation model are followed by a batch normalization layer and a Rectified Linear Unit (ReLU) activation.

2.1.4 Computational efficiency

Many of the state-of-the-art segmentation models are optimized for low to medium-resolution images. This is often the direction for not only lowering computational time but also due to resource limitations, such as GPU memory constraints. To illustrate the challenges arising from high-resolution segmentation, we consider an example of a 1080×1920 image. We estimate that the size of a forward/backward pass for a single image using a U-Net++ model with a ResNeSt50d backbone is approximately 16 GB. Besides a few data centers and industrial GPUs, this VRAM requirement has made the network impossible to train on almost all available consumer GPUs with very few exceptions. A simpler U-Net segmentation model with a ResNet50 backbone would instead require 8.4 GB of GPU memory, which is still very challenging, even if we omit batch training. On the other hand, using the U-Net++(ResNeSt50d) model with a 288×480 image would require an estimate of 1.3 GB of VRAM. Our TRS-Net, with its more advanced internal segmentation model, requires 1.6 GB of GPU memory for passing a full 1080×1920 image, a slight difference from the low-resolution model, making it accessible to train on a wide range of hardware.

TRS-Net is also significantly more computationally efficient in terms of training and inference time. Given the 1080×1920 image, the total multiply–accumulate (MAC) for TRS-Net is estimated at 133 billion. On the other hand, using U-Net++ with a ResNeSt backbone directly on the high-resolution image would result in an estimated 1840 billion MAC. It should be noted that there can be insignificant changes in the MAC units based on the number of predicted classes.

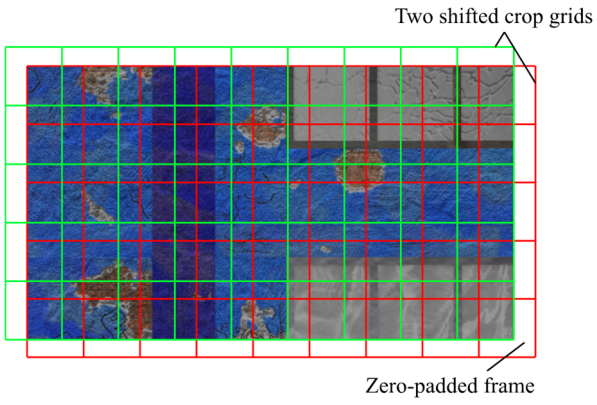


Fig. 7 Augmented inference in DmgFormer

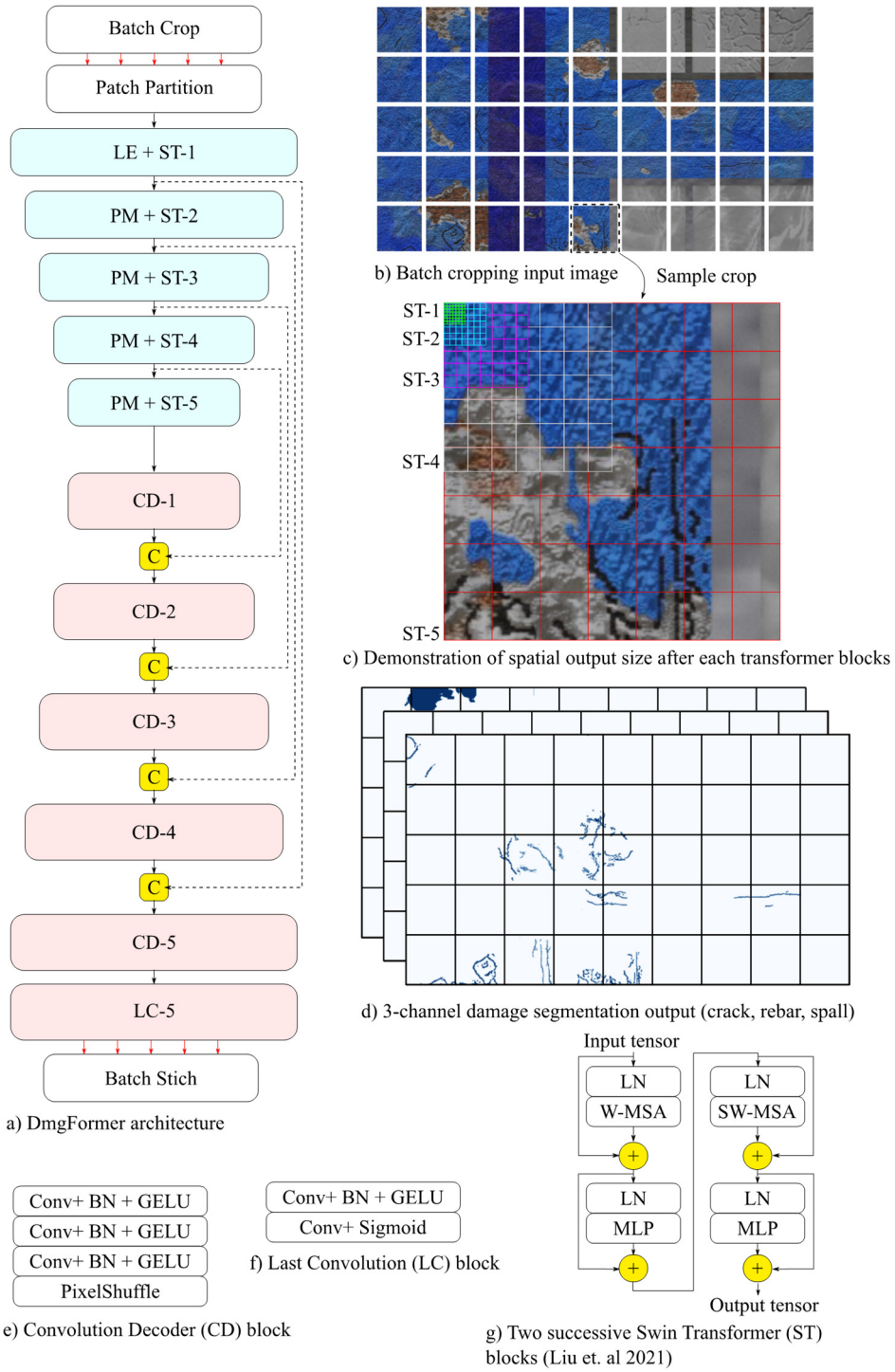


Fig. 8 DmgFormer Architecture. (Conv: 2D convolution, BN: batch normalization, GELU: Gaussian error linear unit, LN: Layer normalization, MLP: multi-layer perceptron, SW/W-MSA: regular and shifted windowed multi-head self-attention, LE: linear embedding, PM: patch merging (Liu *et al*. 2021))

2.2 DmgFormer for damage segmentation

The main challenge in damage segmentation is that the loss of information is often significant after the downsizing of images. It is possible that after recovering the original resolution, pixel information on damage patterns such as thin cracks or exposed rebars is lost. This issue is more critical when the camera location is distant from the building facade. An example of this phenomenon is illustrated in Fig. 5.

While the downsampler and upsampler networks presented earlier might alleviate this problem, we also investigated a partially loss-less approach using the full resolution images. To this end, each image is exploded into a grid of smaller crops. Cropping has both pros and cons for the segmentation tasks. For example, in component

Table 3 DmgFormer (S/L) architecture specification

Stage	DS rate (output size)	Attn. MLP dim.	# STL[(a)]	# Attn. head	Decoder feature dim.
1	2× (112×112)	48/64	2/2	3/4	24/32
2	4× (56×56)	96/128	2/2	6/4	96/124
3	8× (28×28)	192/256	2/2	8/8	180/240
4	16× (14×14)	384/512	6/18	12/16	336/448
5	32× (7×7)	768/1024	2/2	24/32	576/768

[(a)] # STL: Number of Swin Transformer layers

Table 4 Mean testing performance metrics for components and damage state segmentation

	Components				Damage state			
	Precision	Recall	F1-score	IoU	Precision	Recall	F1-score	IoU
Internal TRS-Net (low resolution)	99.40	99.53	99.47	98.94	97.42	97.31	97.36	94.93
Internal TRS-Net + resizing	97.08	96.72	96.89	94.10	96.47	95.50	95.97	92.39
TRS-Net	99.56	99.56	99.56	99.13	98.60	97.83	98.21	96.52

[(a)] # STL: Number of Swin Transformer layers

detection, some context information can be lost if the whole building, including beams, walls, etc., is not present in the model input. Our experiments showed that for damage segmentation, this loss of information due to cropping is insignificant. Nevertheless, cropping results in a) substantial computational gain that made it possible to train more advanced and deeper neural network models for damage segmentation, and b) having high-resolution damage patterns, especially for cracks.

The 1920×1080 images are initially zero-padded (2016×1120) and subsequently cropped into a 5×9 grid of 224×224 RGB images. The deep learning model will receive a batch of these crops for segmentation. As an augmentation technique, we will add random noise translation to the centroid of each crop during training. Perimeter crops that fall outside the vertical or horizontal borders of the padded image are shifted to the border with respect to the direction of trespassing. Fig. 6 demonstrates this process.

2.2.1 Transformers

The deep learning model for damage segmentation benefits from a transformer backbone architecture. Transformers were initially introduced in natural language processing (Vaswani *et al*. 2017) and are currently state of the art in speech recognition (Zhang *et al*. 2020). The core module in a transformer is the multi-head attention that enables effective learning from a sequence. Vision Transformer (ViT) is a more recent type of deep learning architecture that is currently competing with well-established convolutional neural networks (CNN) (Dosovitskiy *et al*. 2021, Liu *et al*. 2022). In fact, the family of ViTs has dominated the top-10 in many benchmark datasets such as ImageNet and ADE20K (PapersWithCode 2022). We introduce DmgFormer, a customized transformer model for damage segmentation (Fig. 8).

The most common and successful examples of segmentation networks include but are not limited to encoder-decoder CNNs such as U-Net, FCN, and DeepLabv3. The presence of several connections between the encoder and decoder modules are critical reasons for better performance in such designs, especially to recover fine segmentation details (Chen *et al*. 2017, Jégou *et al*. 2017). Despite the powerful learning capacity of ViTs as a backbone for classification, building such connections is challenging in typical transformer architectures.

Swin Transformer (Liu *et al*. 2021) is an elegant network design that builds hierarchical feature maps using Patch Merging (PM) as the network gets deeper (Fig. 8). It has linear computational complexity compared to the original ViTs since the self-attention mechanism is computed locally. As demonstrated in Fig. 8(a), we have modified the transformer backbone by increasing the patch size to two instead of four and having five Swin Transformer (ST) blocks. The intuition behind this modification is that for tasks such as crack segmentation, higher precision might be necessary due to the thinner presence of "damage" pixels in the RGB images. Additionally, the five decoder CNN blocks are designed to be consistent with the feature map tensors extracted after each ST block.

Small (S) and Large (L) variants of DmgFormer are considered depending on the initial embedding dimensionality of the attention blocks. Details of the two architectures are presented in Table 3.

2.2.2 Augmented inference

It was mentioned earlier that some context information might be lost despite the computational gains by exploding the image into smaller crops. We propose augmented inference as a technique to alleviate this loss of context. The arrangement of pixels in the 5×9 crop grid depends on the padding direction. Three types of zero padding, including right/top, left/bottom, and middle, were considered for each direction, resulting in a maximum of eight different sets of grid crops. The augmented inference is conducted by

feeding the same 1080p image to the model with different paddings and taking the average of model probabilities. This process is shown in Fig. 7.

3. Case study: QuakeCity dataset

The QuakeCity dataset, released as part of the 2nd International Competition for Structural Health Monitoring (IC-SHM 2021), contains simulated UAV-captured images of buildings that have suffered earthquake-induced damage. The surface damage textures are produced using physics-based graphics models (PBGM) (Hoskere *et al*. 2022). Each 1920×1080 RGB image is associated with five masks including components, components damage states, cracks, spalling, and exposed rebar. The component segmentation includes seven different labels such as walls, beams, and window frames. The damage state segmentation includes four labels representing different levels of damage severity, starting at "No damage" and up to "Severe damage". The dataset contains 3,805 and 1,004 images for training and testing respectively. Since the labels are not provided for the main test set at the time of writing, the original training portion is randomly split into 0.8, 0.1, and 0.1 subsets to further validate and test the neural network models. From this point forward, the testing set refers to the 0.1 splits mentioned earlier unless specified otherwise.

3.1 Implementation

All models are created using PyTorch (Paszke *et al*. 2019), considering its flexibility for custom architecture designs. The initial experiments for TRS-Net were conducted on a desktop equipped with an NVIDIA GTX 1080 with 8 GB of memory, an Intel i7-8700K, and 32 GB of RAM. DmgFormer experiments were also conducted on a workstation equipped with an NVIDIA Titan V GPU with 12 GB of memory, an Intel Core i9 10980-XE, and 64 GB of RAM. Some hyperparameter tunings were conducted using the University at Buffalo's Center for Computational Research (UB-CCR) NVIDIA TESLA GPUs with 16/32 GB of memory.

For TRS-Net, we implement data augmentation techniques during training in all our experiments which are composed of multiple image transforms such as random horizontal flip, zoom, and perspective (Buslaev *et al*. 2020). It also includes random color manipulations such as brightness, gamma, and saturation. We found that data augmentation techniques can help reduce overfitting and help boost the quality of predictions, especially for unseen structures in the main unlabeled test split. TRS-Net model is trained using an Adam optimizer (Kingma and Ba 2014) with an initial learning rate of 1E-3 and exponential decay rates for the 1st and 2nd-moment estimates of 0.9 and 0.999, respectively. We also use a scheduler that further reduces the learning rate if no improvements are observed to a minimum of 1E-5. For the loss function, we use focal loss (Lin *et al*. 2017) with $\alpha = 0.25$ and $\gamma = 2.0$. The model parameters corresponding to the lowest validation loss are used for inference. During experiments, it was found that adopting a greedy layer-wise training strategy improves the training of TRS-Net for components and damage state segmentation (Hinton *et al*. 2006). We initially train the internal segmentation model on downsized images and masks. Then, using the obtained parameters to initialize the internal model, we add the outer encoder-decoder to fine-tune the complete model for high-resolution images. The weights of the backbone ResNeSt network were also initialized using the ImageNet pre-trained weights available in the original authors' PyTorch implementation.

DmgFormers were trained using Adam optimizer with 5 epochs of learning rate warming up to a maximum learning rate of 2E-4 and further decreasing with a cosine function until 300 epochs. Each batch loaded two images and randomly shuffled the grid crops into smaller sub-batches during training to fit the GPU memory. Focal loss with $\alpha = 0.6$ and $\gamma = 2$ was selected based on several experiments. Additionally, and for the sake of comparison, we have trained models using the internal segmentation model of TRS-Net on crack, spall, and exposed rebar segmentation, each trained independently (binary segmentation). To reduce the effect of data imbalance, especially for exposed rebar segmentation, we added a final layer of augmentation during training that crops a random area that has the mask, if it exists in the image. Otherwise, it results in a random crop. These crops have a size of 480×270, thus compatible with the internal segmentation model.

3.2 Results

The first part is evaluating TRS-Net for high-resolution components and damage state segmentation. For each of these two tasks, we have three tests. First, the internal network of TRS-Net is trained and tested on interpolated downsized images and masks. Second, we attach a uniform, non-trainable upsampler and downsampler to the internal network's stem and head and test on high-resolution images. Finally, we train and test our proposed TRS-Net on high-resolution images and compare the results with previous iterations.

The testing results for all six iterations are shown in Table 4. It can be observed that using a uniform sampler (such as nearest-neighbor) for resizing images and masks deteriorates the prediction quality compared to the low-resolution version. On the other hand, TRS-Nets, with their trainable upsampler and downsampler, not only match the low-resolution models but also improve the mask prediction qualities as demonstrated by all performance metrics in the two tasks. This is due to DCN's ability to preserve important features of the original image, such as edges and fine cracks, which can be helpful in the internal segmentation model. Class-wise performance metrics for TRS-Net are shown in Tables 5 and 6. Overall, the performance metrics are relatively close, except for the "no-damage" state in the damage state segmentation task, possibly due to the limited frequency of this label in the dataset. We also experimented with using a Swin-B backbone for TRS-Net and besides increasing the computation time by approximately 40%, the results were inferior to TRS-Net with a ResNeSt backbone resulting in about a mean IoU of only 89.5% at low-resolution.

Table 5 Class-wise testing performance metrics for components segmentation (TRS-Net)

	Wall	Beam	Column	Window frame	Window pane	Balcony	Slab	Ignore
Precision	99.81	99.52	99.75	98.95	99.8	99.84	98.86	99.98
Recall	99.82	99.59	99.75	98.86	99.78	99.83	98.85	99.96
F1-score	99.81	99.56	99.75	98.91	99.79	99.83	98.85	99.97
IoU	99.63	99.12	99.5	97.84	99.59	99.67	97.74	99.94

Table 6 Class-wise testing performance metrics for damage-state segmentation (TRS-Net)

	No damage	Light damage	Moderate damage	Severe damage	Ignore
Precision	96.98	98.5	99.23	98.42	99.85
Recall	94.07	98.92	99.22	97.12	99.81
F1-score	95.50	98.71	99.22	97.77	99.83
IoU	91.40	97.45	98.46	95.63	99.66

Table 7 Testing performance metrics for damage segmentation

	Precision			F1-score		
	Crack	Rebar	Spall	Crack	Rebar	Spall
DmgFormer-S	85.46	85.56	97.75	86.06	85.17	98.09
DmgFormer-S (AI-4)	85.96	87.20	97.87	86.21	85.93	98.16
DmgFormer-S (AI-8)	86.09	87.07	97.89	86.26	85.94	98.17
DmgFormer-L	85.82	88.76	97.94	86.82	87.35	98.27
DmgFormer-L (AI-4)	86.25	89.84	98.06	86.98	87.92	98.34
DmgFormer-L (AI-8)	86.41	89.97	98.10	87.05	88.08	98.37
TRS-Net (internal network)	82.69	88.01	96.72	84.87	86.48	97.03
	Recall			IoU		
	Crack	Rebar	Spall	Crack	Rebar	Spall
DmgFormer-S	86.67	84.8	98.44	75.53	74.18	96.25
DmgFormer-S (AI-4)	86.47	84.7	98.45	75.77	75.33	96.38
DmgFormer-S (AI-8)	86.43	84.84	98.45	75.84	75.35	96.4
DmgFormer-L	87.83	85.99	98.60	76.70	77.54	96.60
DmgFormer-L (AI-4)	87.72	86.08	98.62	76.96	78.44	96.74
DmgFormer-L (AI-8)	87.70	86.27	98.63	77.07	78.70	96.78
TRS-Net (internal network)	87.17	85.01	97.35	73.72	76.18	94.24

The testing results of cracks, spalling, and exposed rebars segmentation using DmgFormer are given in Table 7. The testing performance of the internal portion of TRS-Net is also included for comparison. It can be observed that preserving the original resolution by cropping yields higher fidelity segmentation masks. Furthermore, by using 4 and 8 iterations of augmented inference (AI-4 & 8), performance is boosted at the cost of increased inference times. Our experiments show that training a single model that simultaneously predicts rebar, crack, and spalling has superior performance over three single models. One hypothesis about this phenomenon is that a model that learns to predict all classes simultaneously learns its features maps to differentiate between cases where damage classes could have the same appearances, especially for cracks and spalling. We also found that less than 1% of the dataset has rebar pixels, and its frequency is approximately 30 times less than cracks and spalling. Examples of the framework mask predictions for all five tasks for the QuakeCity test set are depicted in Fig. 9.

4. Conclusions

Visual structural inspections are entering a new era with the latest advances in sensing and computing technology accompanied by the giant progress in AI research. UAVs can equip building owners and inspectors with thousands of high-resolution frames from a building façade. Effectively processing this information for autonomous inspections requires high-performance and computationally cost-effective models that ultimately contribute to UAV

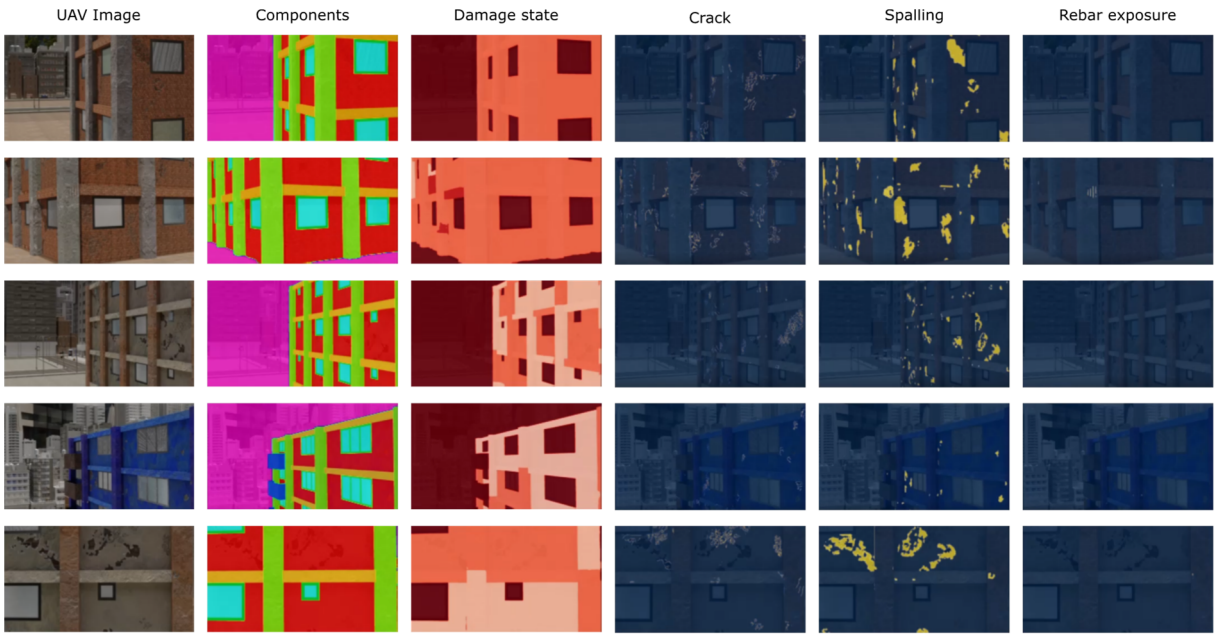


Fig. 9 Test prediction examples from the QuakeCity test set

navigation and damage identification. State-of-the-art deep learning vision models are moving toward larger models with massive parameters and are not optimized for SHM vision tasks. This poses a great challenge for autonomous inspection tasks. Real-time inference in detecting structural components and their severity is critical for optimal UAV guidance. Furthermore, high-resolution segmentation is a must in tasks such as identifying thin cracks. Therefore, systems that can accurately, yet efficiently, provide autonomous visual inspections are highly desired. The QuakeCity dataset highlights these challenges by containing several inherently different tasks handling full HD images.

In this work, we argue that component type and damage state segmentation rely on global context. In contrast, damage segmentation, such as cracks, spalling, and rebars, is highly sensitive to preserving the input image resolution. We approached this problem by proposing two custom deep learning frameworks namely: TRS-Net and DmgFormer. TRS-Net utilizes an internal segmentation model based on the ResNeSt backbone and U-Net++ decoder in addition to an outer encoder-decoder of trainable down and upsampling networks. It was found that the trainable resizing modules significantly improve the model's efficiency while minimizing the loss of information during resizing in components and damage state segmentation. Cracks, spalling, and exposed rebar segmentations are conducted with DmgFormer which is equipped with a Swin transformer backbone and a convolutional decoder. It is shown that using crops of the original image resolution with the proposed augmented inference technique can result in better segmentation metrics.

In our visual inspection framework, we attempted to not only create a balance between computational efficiency and prediction accuracy but also cater to the degree of attention to local and global context that is given by each task. Therefore, our study produced a two-model scheme, one that focuses on the global context, and the other preserves local details. The inconvenience of having two models to achieve autonomous visual inspection can be seen as a disadvantage to this method despite the performance gain. However, we believe that our findings provide a first step towards developing a unified model that can learn to focus on both global and local semantics based on the given task.

References

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

Agyemang, I.O., Zhang, X., Mensah, I.A., Mawuli, B.C., Agbley, B.L.Y. and Arhin, J.R. (2021), "Enhanced deep convolutional neural network for building component detection towards structural health monitoring", *Proceedings of the 4th International Conference on Pattern Recognition and Artificial Intelligence* (*PRAI*), Yibin, China, August. https://doi.org/10.1109/PRAI53619.2021.9551102

Buslaev, A., Iglovikov, V.I., Khvedchenya, E., Parinov, A., Druzhinin, M. and Kalinin, A.A. (2020), "Albumentations: fast and flexible image augmentations", *Information*, **11**(2), 125. https://doi.org/10.3390/info11020125

Chen, L.-C., Papandreou, G., Kokkinos, I., Murphy, K. and Yuille, A.L. (2017), "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs", *IEEE Transact. Pattern Anal. Mach. Intell.*, **40**(4), 834-848. https://doi.org/10.1109/TPAMI.2017.2699184

Dorafshan, S., Thomas, R.J. and Maguire, M. (2018), "Fatigue crack detection using unmanned aerial systems in fracture critical inspection of steel bridges", *J. Bridge Eng.*, **23**(10), 04018078. https://doi.org/10.1061/(ASCE)BE.1943-5592.0001291

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J. and Houlsby, N. (2021), "An Image is Worth 16×16 Words: Transformers for Image Recognition at Scale", *Proceedings of 2021 International Conference on*

Learning Representations (ICLR).
Guo, J., Wang, Q. and Li, Y. (2021), "Semi-supervised learning based on convolutional neural network and uncertainty filter for façade defects classification", *Comput.-Aided Civil Infrastr. Eng.*, **36**(3), 302-317. https://doi.org/10.1111/mice.12632
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.
He, T., Zhang, Z., Zhang, H., Zhang, Z., Xie, J. and Li, M. (2019), "Bag of tricks for image classification with convolutional neural networks", *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*.
Hinton, G.E., Osindero, S. and Teh, Y.-W. (2006), "A fast learning algorithm for deep belief nets", *Neural Computat.*, **18**(7), 1527-1554. https://doi.org/10.1162/neco.2006.18.7.1527
Hoskere, V., Narazaki, Y., Hoang, T. and Spencer Jr, B. (2017), "Vision-based structural inspection using multiscale deep convolutional neural networks", *Proceedings of the 3rd Huixian International Forum on Earthquake Engineering for Young Researchers*, Urbana, IL, USA. https://doi.org/10.48550/arXiv.1805.01055
Hoskere, V., Narazaki, Y., Hoang, T.A. and Spencer Jr, B. (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**(5), 757-773. https://doi.org/10.1007/s13349-020-00409-0
Hoskere, V., Narazaki, Y. and Spencer Jr, 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
Hou, X., Zeng, Y. and Xue, J. (2020), "Detecting structural components of building engineering based on deep-learning method", *J. Constr. Eng. Manag.*, **146**(2), 04019097. https://doi.org/10.1061/(ASCE)CO.1943-7862.0001751
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.
IC-SHM (2021), Structures and Artificial Intelligence Lab. Retrieved 2022/04/30 from https://sail.cive.uh.edu/ic-shm2021/
Jahanshahi, M.R. and Masri, S.F. (2012), "Adaptive vision-based crack detection using 3D scene reconstruction for condition assessment of structures", *Automat. Constr.*, **22**, 567-576. https://doi.org/10.1016/j.autcon.2011.11.018
Jégou, S., Drozdzal, M., Vazquez, D., Romero, A. and Bengio, Y. (2017), "The one hundred layers tiramisu: Fully convolutional densenets for semantic segmentation", *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 11-19.
Katharopoulos, A. and Fleuret, F. (2019), "Processing megapixel images with deep attention-sampling models", *Proceedings of the 36th International Conference on Machine Learning*, pp. 3282-3291.
Kingma, D.P. and Ba, J. (2014), "Adam: A method for stochastic optimization", arXiv preprint arXiv:1412.6980. https://doi.org/10.48550/arXiv.1412.6980
Li, L., Wang, Q., Zhang, G., Shi, L., Dong, J. and Jia, P. (2018), "A method of detecting the cracks of concrete undergo high-temperature", *Constr. Build. Mater.*, **162**, 345-358. https://doi.org/10.1016/j.conbuildmat.2017.12.010
Liang, X. (2019), "Image-based post-disaster inspection of reinforced concrete bridge systems using deep learning with Bayesian optimization", *Comput.-Aided Civil Infrastr. Eng*, **34**(5), 415-430. https://doi.org/10.1111/mice.12425
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, Z., Cao, Y., Wang, Y. and Wang, W. (2019), "Computer vision-based concrete crack detection using U-net fully convolutional networks", *Automat. Constr.*, **104**, 129-139. https://doi.org/10.1016/j.autcon.2019.04.005
Liu, Y.F., Nie, X., Fan, J.S. and Liu, X.G. (2020), "Image-based crack assessment of bridge piers using unmanned aerial vehicles and three-dimensional scene reconstruction", *Comput.-Aided Civil Infrastr. Eng.*, **35**(5), 511-529. https://doi.org/10.1111/mice.12501
Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S. and Guo, B. (2021), "Swin transformer: Hierarchical vision transformer using shifted windows", arXiv preprint arXiv:2103.14030. pp. 10012-10022
Liu, Z., Mao, H., Wu, C.-Y., Feichtenhofer, C., Darrell, T. and Xie, S. (2022), "A convnet for the 2020s". Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 11976-11986.
Munawar, H.S., Hammad, A.W., Haddad, A., Soares, C.A.P. and Waller, S.T. (2021), "Image-based crack detection methods: A review", *Infrastructures*, **6**(8), 115. https://doi.org/10.3390/infrastructures6080115
Narazaki, Y., Hoskere, V., Hoang, T.A., Fujino, Y., Sakurai, A. and Spencer Jr, B.F. (2020), "Vision-based automated bridge component recognition with high-level scene consistency", *Comput.-Aided Civil Infrastr. Eng.*, **35**(5), 465-482. https://doi.org/10.1111/mice.12505
Narazaki, Y., Hoskere, V., Chowdhary, G. and Spencer Jr, B.F. (2022), "Vision-based navigation planning for autonomous post-earthquake inspection of reinforced concrete railway viaducts using unmanned aerial vehicles", *Automat. Constr.*, **137**, 104214. https://doi.org/10.1016/j.autcon.2022.104214
PapersWithCode (2022), Semantic Segmentation. https://paperswithcode.com/task/semantic-segmentation
Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N. and Antiga, L. (2019), "Pytorch: An imperative style, high-performance deep learning library", *Adv. Neural Inform. Process. Syst.*, **32**, 8026-8037.
Sajedi, S.O. and Liang, X. (2019), "A convolutional cost-sensitive crack localization algorithm for automated and reliable RC bridge inspection", In: *Risk-Based Bridge Engineering*, pp. 229-235.
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
Shi, W., Caballero, J., Huszár, F., Totz, J., Aitken, A.P., Bishop, R., Rueckert, D. and Wang, Z. (2016), "Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network", *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1874-1883.
Talab, A.M.A., Huang, Z., Xi, F. and HaiMing, L. (2016), "Detection crack in image using Otsu method and multiple filtering in image processing techniques", *Optik*, **127**(3), 1030-1033. https://doi.org/10.1016/j.ijleo.2015.09.147
Tan, M. and Le, Q. (2019), "Efficientnet: Rethinking model scaling for convolutional neural networks", *International Conference on Machine Learning*, pp. 6105-6114.
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
Teng, S., Liu, Z., Chen, G. and Cheng, L. (2021), "Concrete crack detection based on well-known feature extractor model and the YOLO_v2 network", *Appl. Sci.*, **11**(2), 813. https://doi.org/10.3390/app11020813

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł. and Polosukhin, I. (2017), "Attention is all you need", *Adv. Neural Inform. Process. Syst.*

Wightman, R. (2019), PyTorch Image Models, GitHub. https://github.com/rwightman/pytorch-image-models

Wu, Y., Qin, Y., Qian, Y., Guo, F., Wang, Z. and Jia, L. (2022), "Hybrid deep learning architecture for rail surface segmentation and surface defect detection", *Comput.-Aided Civil Infrastr. Eng.*, **37**(2), 227-244. https://doi.org/10.1111/mice.12710

Yakubovskiy, P. (2020), Segmentation Models Pytorch. GitHub. https://github.com/qubvel/segmentation_models.pytorch

Zhang, A., Wang, K.C., Li, B., Yang, E., Dai, X., Peng, Y., Fei, Y., Liu, Y., Li, J.Q. and Chen, C. (2017), "Automated pixel-level pavement crack detection on 3D asphalt surfaces using a deep-learning network", *Comput.-Aided Civil Infrastr. Eng.*, **32**(10), 805-819. https://doi.org/10.1111/mice.12297

Zhang, K., Cheng, H. and Zhang, B. (2018), "Unified approach to pavement crack and sealed crack detection using preclassification based on transfer learning", *J. Comput. Civil Eng.*, **32**(2), 04018001. https://doi.org/10.1061/(ASCE)CP.1943-5487.0000736

Zhang, H., Wu, C., Zhang, Z., Zhu, Y., Lin, H., Zhang, Z., Sun, Y., He, T., Mueller, J. and Manmatha, R. (2020), "Resnest: Split-attention networks", arXiv preprint arXiv:2004.08955.

Zhang, H., Wu, C., Zhang, Z., Zhu, Y., Lin, H., Zhang, Z., Sun, Y., He, T., Mueller, J. and Manmatha, R. (2022), "Resnest: Split-attention networks", *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2736-2746.

Zhang, Y., Qin, J., Park, D.S., Han, W., Chiu, C.-C., Pang, R., Le, Q.V. and Wu, Y. (2020), "Pushing the limits of semi-supervised learning for automatic speech recognition", arXiv preprint arXiv:2010.10504. https://doi.org/10.48550/arXiv.2010.10504

Zheng, Y., Gao, Y., Lu, S. and Mosalam, K.M. (2022), "Multistage semisupervised active learning framework for crack identification, segmentation, and measurement of bridges", *Comput.-Aided Civil Infrastr. Eng.*, **37**(9), 1089-1108. https://doi.org/10.1111/mice.12851

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. https://doi.org/10.1007/978-3-030-00889-5_1

Zhou, Z., Zhang, J. and Gong, C. (2022), "Automatic detection method of tunnel lining multi-defects via an enhanced You Only Look Once network", *Comput.-Aided Civil Infrastr. Eng.*, **37**(6), 762-780. https://doi.org/https://doi.org/10.1111/mice.12836