Journal Title
XX(X):1–15

DOI: 10.1177/ToBeAssigned
www.sagepub.com/
SAGE

A semi-supervised self-training method to develop assistive intelligence for segmenting multiclass bridge elements from inspection videos

Muhammad Monjurul Karim[1], Ruwen Qin[1], Zhaozheng Yin[2] and Genda Chen[3]

Abstract

Bridge inspection is an important step in preserving and rehabilitating transportation infrastructure for extending their service lives. The advancement of mobile robotic technology allows the rapid collection of a large amount of inspection video data. However, the data are mainly images of complex scenes, wherein a bridge of various structural elements mix with a cluttered background. Assisting bridge inspectors in extracting structural elements of bridges from the big complex video data, and sorting them out by classes, will prepare inspectors for the element-wise inspection to determine the condition of bridges. This paper is motivated to develop an assistive intelligence model for segmenting multiclass bridge elements from inspection videos captured by an aerial inspection platform. With a small initial training dataset labeled by inspectors, a Mask Region-based Convolutional Neural Network (Mask R-CNN) pre-trained on a large public dataset was transferred to the new task of multiclass bridge element segmentation. Besides, the temporal coherence analysis attempts to recover false negatives and identify the weakness that the neural network can learn to improve. Furthermore, a semi-supervised self-training (S^3T) method was developed to engage experienced inspectors in refining the network iteratively. Quantitative and qualitative results from evaluating the developed deep neural network demonstrate that the proposed method can utilize a small amount of time and guidance from experienced inspectors (3.58 hours for labeling 66 images) to build the network of excellent performance (91.8% precision, 93.6% recall, and 92.7% f1-score). Importantly, the paper illustrates an approach to leveraging the domain knowledge and experiences of bridge professionals into computational intelligence models to efficiently adapt the models to varied bridges in the National Bridge Inventory.

Keywords

Transfer learning, temporal coherence, self-training, active learning, human-in-the-loop, AI-human collaboration, bridge inspection, multiclass segmentation

Introduction

The U.S National Bridge Inventory has over 600,000 highway bridges. 39% of these bridges are over 50 years old, and almost 9% are structurally deficient and require significant repair[1]. Rehabilitation, maintenance, and rebuilding efforts are necessary for preserving the transportation infrastructure throughout the United States. For example, National Bridge Inspection Standards require that each bridge should be inspected every two years to ensure no cracks, rust, or other damages[2]. The conventional bridge inspection requires a crew of inspectors, heavy equipment with a lifting capability, access to dangerous heights, and the closure of the road during the time of inspection. These make the bridge inspection one of the most dangerous and costly operations in the state Departments of Transportation. Results of the visual inspection are inaccurate and vary largely among different inspectors although the image-reference approach is developed to guide the inspection[3].

Research has taken place to develop safer and more efficient bridge inspection methods. Some adopted a completely manual approach for the bridge routine inspection, which requires a large number of inspector hours[4] and inspection results vary largely among inspectors. To make inspection faster, cheaper, safer, more objective, and less interruptive to traffic, methods to automate the bridge inspection have been developed. Recently, mobile robots such as Unmanned Aerial Vehicles (UAVs) have been proven to be very helpful in dangerous, dull, or dirty applications[5]. Collecting inspection video data using aerial platforms reduces or eliminates the labor-intensive onsite inspection process and allows inspectors to assess bridges from a safer location. Yet, the use of robotic inspection platforms has solved just part of the above-discussed issues,

[1]Department of Civil Engineering, Stony Brook University, Stony Brook, NY 11794, USA
[2]AI Institute, Department of Computer Science, Department of Biomedical Informatics, Stony Brook University, Stony Brook, NY 11794, USA
[3]Department of Civil, Environmental and Architectural Engineering, Missouri University of Science and Technology, Rolla, MO 65409

Corresponding author:
Ruwen Qin, Department of Civil Engineering, Stony Brook University, Stony Brook, NY 11794, USA.
Email: ruwen.qin@stonybrook.edu

efficient, reliable analysis of inspection video is another important task.

Letting inspectors watch the collected videos for hours and days are inefficient. It is desired that a tool can be developed to assist inspectors in extracting structural elements from the inspection videos and sorting them out by classes. Given a such tool, inspectors can concentrate on the element-wise inspection. Besides, the rating of a bridge needs to be provided by a comprehensive assessment that evaluates the impact of defects on specific elements of the bridge[3,6,7]. This requires to spatially relate detected defects with bridge elements where the defects are located. The above-mentioned approach to the bridge condition evaluation suggests that an important step of analyzing the inspection video data is to extract and index images of bridge elements. After that, defect evaluation and interpretation will take place.

Extracting structural elements from the inspection videos and sorting them out by classes is a very challenging task for practitioners. On the one hand, there could be hours of videos that need to be analyzed for every individual bridge of inspection. Watching hours of video to locate the desired regions of interest is very cumbersome work for a human. Humans are prone to fatigue. Studies have shown that the human visual inspection accuracy declines easily in dull, endlessly routine job[8,9]. The inspector could easily miss elements in big video data, left there without an examination. Fatigue and boredom developed from repetitively watching hours of video data induce the bias in assessing the bridge elements and evaluating the condition of the whole bridge. On the other hand, bridge inspection videos captured by aerial inspection platforms are mainly images of complex scenes, wherein a bridge of various structural elements mix with a cluttered background. Assisting inspectors in analyzing the big complex video data is greatly desired to improve their job efficiency. The development of sensing technology and deep learning methods has significantly advanced the image analysis for defect detection[10–16]. Yet, methods to create deep learning models for defect detection and classification are not directly applicable to the research problem of this paper for various reasons. For example, many of the models require to take close-up images in a nearly uniform testing background where defects are relatively large and clear to analyze. Although deep learning models for segmenting multiclass objects from images are well developed in computer vision, extracting multiclass bridge elements from inspection videos captured by aerial robotic platforms is not a completely solved problem.

A few studies have developed a strong base for infrastructure component recognition using computer vision[17,18]. Extracting bridge structural elements from videos captured by aerial inspection platforms are facing additional challenges[19]. These include, but not limited to, motion blur, partial or full occlusion, illumination variation, background variation, and so on. So far, some studies[20–23] have reported their successful experiences, for example, utilizing the temporal information of objects in video data. But the additional computational cost is expensive. The high accuracy of deep learning models for multiclass object detection and segmentation relies on large-scale dense annotations for model training. Yet annotating a huge amount of training data for bridge inspection is not only labor-intensive but expensive as it needs the knowledge of domain experts[24]. To truly assist bridge inspectors in their jobs, the burden of data annotation should not be completely passed to them. The efforts that domain experts, such as inspectors, contribute to the deep learning model development must be well controlled and best utilized. The strict budget for inspector-annotated training data and the high requirement on model performance motivate the combination of self-training and active learning to create a new model training approach, which are delineated in the next section.

This paper proposes a cost-effective method to create an assistive intelligence model for detecting and segmenting multiclass structural elements from bridge inspection videos captured by an aerial inspection platform. Achieved job efficiency and the quality of the model let inspectors truly benefit from the technology advancement in their jobs. The assistive intelligence model is not an artificial intelligence model isolated from users. Instead, inspectors provide their expertise to guide the development of a deep neural network, which assures the network quickly converges to a satisfactory tool for assisting themselves in analyzing the videos of any intended bridge of inspection. Filling the gaps identified in this paper, the proposed method has anticipated technical contributions in three-fold: (i) a quick transfer of an existing deep learning network to the task of detecting and segmenting multiclass structural elements from bridge inspection videos, (ii) the use of a lightweight temporal coherence analysis to recover false negatives and identify weakness that the network can learn to improve, and (iii) the development of a semi-supervised self-training (S^3T) algorithm that keeps human-in-the-loop to efficiently refine the deep neural network iteratively.

The remainder of this paper is organized as the following. The related work is discussed in the next section. Then, the proposed method to create the assistive intelligence model is delineated. After that, results from evaluating the proposed method and the developed model are discussed. In the end, conclusions and future work are summarized.

Literature Review

Being an important step before the detailed damage assessment, extracting regions of interest from inspection video data is receiving attention from the bridge health monitoring community. A few studies have developed a strong base for infrastructure component recognition using computer vision. For example, Narazaki et al.[17] used multi-scale convolutional neural networks to perform the pixel-wise classification and smoothed the segmentation result using conditional random forest. They used the scene classification result to help reduce false positives of bridge components in complex scene images. Recently, authors from the same research group[18] further examined two semantic segmentation algorithms and three approaches to integrating a scene classifier and a bridge component classifier. This study found that the sequential configuration outperforms other configurations if the input is complex scene images. Yeum et al.[19] discussed various difficulties in analyzing inspection video data collected by aerial platforms and proposed a Convolutional Neural Network (CNN)

based approach to locate and extract regions of interest from images before performing the damage detection. The study demonstrated the implementation of the developed neural network in finding candidate image patches of welded joints of the truss structure. It also showed that detecting highly relevant structural elements can greatly reduce the false positive and false negative detection in the following step of damage assessment. Yet, detecting and segmenting multiclass structural elements from inspection videos collected by aerial platforms is still not solved completely.

On multiclass object detection, the Region-based CNN (R-CNN)[25] has shown success in many applications. The R-CNN uses the selective search[26] to generate region proposals to find objects in an image. The Faster R-CNN[27] was proposed to make the R-CNN faster. It offers improvements in both speed and accuracy over its predecessors through the shared computation and the use of a neural network to propose regions. Then, the Mask R-CNN[28], an extension of the Faster R-CNN, was proposed to perform the bounding box regression and the pixel-level segmentation simultaneously. The R-CNN and Mask R-CNN models work well in detecting objects from static images. But results may not be consistent when they process video data. Therefore, the temporal coherence information of objects in successive frames has been introduced to address the issue of inconsistent detection[20–22], wherein the tubelet and optical flow are used to propagate features from one frame to another. Temporal coherence analysis methods in the literature are computationally expensive due to the requirement for repeated motion estimation and feature propagation. Seq-NMS[23] has a modification only in the post-processing phase and, thus, it is faster than others. However, seq-NMS tends to increase the volume of false positives because it neither puts any penalty on false positives nor adds additional constraints to prevent the occurrence.

Creating a deep neural network usually requires a huge amount of annotated data for model training. The manual annotation of data is not only a costly process but often prone to errors. To overcome this issue, transfer learning is introduced to structural damage detection[29–32]. With transfer learning, only a relatively small dataset is needed to refine an existing deep network, which reduces the training time while keeping a good performance. Another way to tackle this annotation problem is to use semi-supervised learning that requires some labeled and some unlabeled training data. Papandreo et al.[33] developed a method requiring a small number of strongly annotated images and a large number of weakly annotated images for training. They used an expectation maximization method to generate the pixel-level annotation from weakly annotated training data. Mittal et al.[34] proposed an approach that relies on adversarial training with a feature matching loss to learn from unlabeled images. Some researchers used the self-training, a wrapper based semi-supervised method[35], which starts training a network with only a few annotated samples and then let the network automatically annotate more training samples. This technique has been applied to a variety of image/video processing applications[36,37] to reduce the effort of human annotation. While most semi-supervised learning methods save the data annotation effort compared to supervised learning, their performance is still not good enough. The primary reason for this challenge is the quality of the automatically annotated data. The inclusion of some samples mislabeled by the network itself may sharply deteriorate the training process.

Recently, a promising approach named active learning has been proposed to reduce the annotation cost for training deep neural networks[38]. The essence of active learning is to train a network by actively selecting training samples from a pool of unlabeled data, which will be labeled by a human annotator to re-train the model. The fundamental assumption underlying this approach is that selecting fewer but informative data and allowing the model to learn with it may achieve greater performance than training the network using a large amount of labeled data. The performance of active learning depends on the samples selected from the unlabeled data. Traditional sampling methods do not always guarantee to provide the most representative training samples in support of active learning because of the dataset diversity and limited knowledge about the dataset. Many strategies have been developed for sampling data from unlabeled data for active learning[38]. Uncertainty sampling is the most widely used strategy, which queries for samples that it is most unsure about. For example, Tian et al.[39] used clustering and the fuzzy-set selection method to choose the most uncertain and informative samples. This method increases the training sample diversity. The use of active learning in image classification has been widely explored[40,41]. However, the potential of active learning is less thoroughly explored in the more complex task of instance segmentation that usually has a relatively higher annotation cost. Morrison et al.[42] considered both spatial and semantic uncertainties of prediction using the dropout sampling. By adding dropout layers to the fully connected layers of the network, multiple times of inference over the same image are made to measure the segmentation uncertainty. A similar approach was adopted by Gal et al.[43] who used multiple forward passes with dropout at the inference (Monte Carlo dropout) to obtain better uncertainty estimates for instance segmentation. Yang et al.[44] trained a set of fully convolutional networks iteratively and estimated uncertainty and similarity from an ensemble of networks to determine new data for annotation. The above-discussed studies demonstrated that determining the most informative samples for active learning is complex. Besides, uncertainty-based approaches can be prone to querying outliers[45]. Simple but effective methods for recommending new data for annotation are greatly desired.

Methodology

The proposed method to create the assistive intelligence model for the multiclass bridge element segmentation is illustrated in Figure 1. First, a pre-trained Mask R-CNN has been chosen. Then, a small set of initial training data, which are annotated by the inspector, is used to fine-tune the network to transfer it for the new task of multiclass bridge element segmentation. The transferred network will be improved iteratively until it achieves the satisfactory performance. The shaded portion in Figure 1 is the iterative process for performance boosting. In each iteration of the

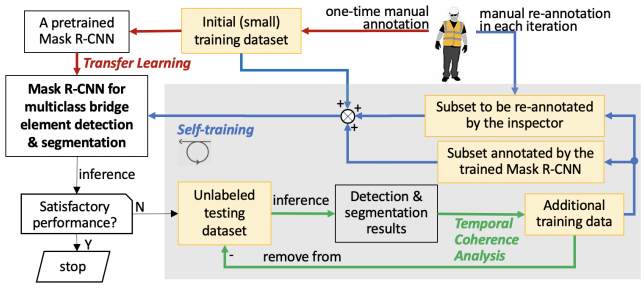


Figure 1. Overview of the proposed S^{3}T method with human-in-the-loop.

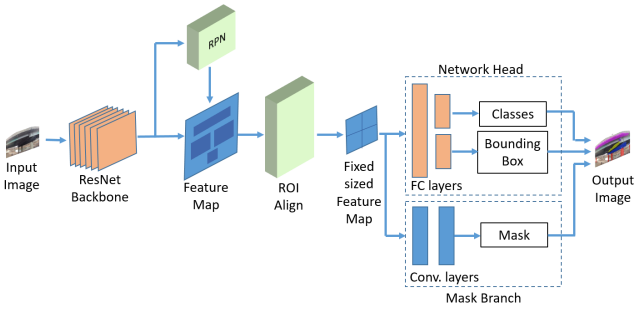


Figure 2. The architecture of Mask R-CNN that performs the detection and segmentation of multiclass bridge elements.

semi-supervised self-training (S^{3}T) with human-in-the-loop, the network that has not reached the satisfied performance will be applied to an unlabeled testing dataset to obtain the detection and segmentation results. Temporal coherence analysis of the results is performed to recover false negative results that are hard data for the network. After that, a small set of the recovered hard data is selected as additional training data that are removed from the unlabeled testing dataset. The additional training data is then split into two subsets, one has been automatically annotated by the network trained from the current iteration and the other subset is manually re-annotated by the inspector. The additional training data along with the initial small training dataset is then used to re-train the Mask R-CNN in the next iteration to boost its performance. The S^{3}T method let the inspector annotate selected samples that the current network failed to detect. Through learning from its weakness, the performance of the network increases quickly after a few iterations.

Adapting the Deep Neural Network to a New Task Through Transfer Learning

This study chose a Mask R-CNN as the tool for detecting and segmenting bridge elements from inspection video data. Figure 2 illustrates the structure of the Mask R-CNN. Video data are input into the network frame by frame following their order on the timeline. The backbone of the network is a feature extractor that generates the feature map of each input image. A region proposal network (RPN) creates proposal boxes named anchors and predicts the possibility of an anchor being a bridge element. Then, the RPN ranks anchors and proposes those most likely containing bridge elements, which are termed RoIs. A layer named Region of Interests Align (RoIAlign) extracts the region of interests (RoIs) from the feature map, aligns them with the input image, and converts them into fixed-size region feature maps. The fixed-size feature maps of RoIs are fed into two independent branches: the network head branch that performs the classification and bounding box generation, and the mask branch that independently generates instance masks. Readers interested in the detail of Mask R-CNN can refer to the work by He et al.[28].

Training the Mask R-CNN for the new task of multiclass bridge element segmentation from the scratch requires a large volume of annotated data to achieve a satisfying prediction accuracy. This task does not have a large volume of annotated data for model training. To obtain high quality annotated data for this task requires the knowledge of professionals in the domain of study. Only bridge inspectors are confident in annotating bridge elements from the inspection videos. In this study, transfer learning is first used to tackle this challenge, which improves learning of the new task by transferring knowledge from a related task that has already been learned[46].

The Mask R-CNN in this study was initialized by adopting the ResNet-50 feature extractor[47] whose weights have been pre-trained on the Microsoft COCO dataset consisting of more than 120,000 labeled images and around 1.5 millions of object instances in 80 categories[24]. Then, transfer learning was used to adapt this feature extractor to the setting of bridge inspection. Specifically, the ResNet-50 was fine-tuned using a small set of training data (T_0) with a portion collected from the intended bridges of inspection. The detail of the fine-tuning process will be presented in the next section.

Temporal Coherence Analysis for Recovering False Negative Results

Mask R-CNN is a static image detector in that it processes individual images independently. When it is applied to frames of a video stream, false negative results are likely to happen due to sudden scale changes, occlusion, or motion blur. This study used the temporal coherence information of objects in successive frames to recover false negative detections and segmentations.

Consider a video clip that consists of a series of N frames, indexed by i. In each frame the network returns M_i objects with segmentations, indexed by j. An object in a frame is highly likely to present in its neighboring frames within a range of displacement. Let, $o_{i,j}$ designate object j in frame i. The center of the bounding box for $o_{i,j}$ is specified by its coordinates $C_{i,j} = (x_{i,j}, y_{i,j})$. In p frames, $C_{i,j}$ may shift to a surrounding pixel within a spatial displacement of $p\Delta d$ where Δd is the maximum displacement between two consecutive frames. Δd is affected by both intrinsic and extrinsic camera parameters, as the Appendix explains. Δd is proportional to the focal length of camera and the maximum displacement of the moving camera between capturing two successive frames of image; it is inversely proportional to the size of pixels in the images taken by the camera and the distance of the camera to the object along the optical axis. The study roughly estimated Δd according to its formula in Eq. (5) using partial information and then improved the estimation experimentally by reviewing the inspection data.A value of 60 pixels was found to be appropriate in this

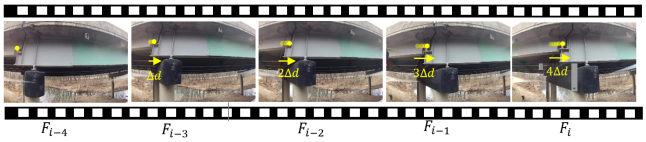


Figure 3. An illustration of spatial displacements.

study. Figure 3 illustrates an example wherein a joint of the bridge in frame $i-4$ is also shown in the succeeding four frames but with displacements.

The algorithm of temporal coherence analysis for recovering false negative results is summarized as the pseudo-code in Algorithm 1 and explained below. The prediction threshold is set as a range $[t_l, t_u]$. An object with a prediction score within this range is possibly a false negative prediction. Let $S_{i,j}$ denote the prediction score for object $o_{i,j}$. The network immediately returns a positive result if $S_{i,j} \geq t_u$ and will not predict any object if $S_{i,j} < t_l$. Let O_i be the set of confidently predicted objects in frame i. The prediction score and the center location of these objects, $\{(S_{i,j}, C_{i,j})|o_{i,j} \in O_i\}$, are the temporal coherence information for analyzing the succeeding k frames. That is, k is the temporal window that defines the range of preceding frames where the temporal coherence analysis searches the same objects as the weakly predicted objects (e.g., they are possibly false negative results) in the current frame. If $t_l \leq S_{i,j} < t_u$, the weakly predicted object $o_{i,j}$ is checked by referring to a pair of preceding successive frames up to $k-1$ times, starting from the nearest pair (frames $i-1$ and $i-2$) to the farthest pair (frames $i-k+1$ and $i-k$). If an object of the same class as $o_{i,j}$ is found in both frames $i-1$ and $i-2$ (i.e., there exists $o_{i-1,j'} \in O_{i-1}$ and $o_{i-2,j''} \in O_{i-2}$ such that $o_{i-1,j'} = o_{i-2,j''} = o_{i,j}$), and the spatial displacements of $o_{i,j}$ from $o_{i-1,j'}$ and $o_{i-2,j''}$ are small, within Δd and $2\Delta d$, respectively, this weakly predicted object is determined as a false negative prediction. The false negative prediction is recovered by adding it to O_i and updating its score to be the average score of $S_{i-1,j'}$ and $S_{i-2,j''}$. Otherwise, $o_{i,j}$ is searched in O_{i-2} and O_{i-3} to determine if it is a false negative result that can be recovered. This search will continue as needed. If $o_{i,j}$ is not found to be a positive result with confidence in the neighboring frames after $k-1$ times of temporal coherence analysis, it will be eliminated from the candidate list. Using a pair of frames as the reference instead of referring to a single frame will make the temporal coherence rule more strict and minimize the risk of progressively propagating a false positive result in a single frame to the target frame.

The appendix has explained factors that impact the choice of Δd. The choice of k is relevant to Δd. Given that Δd is 60 pixels, an object may appear in a sequence of frames because the dimension of images is way larger than 60 pixels. The object only appears in a short video clip and then it disappears because the UAV brings the camera away from the object. If k is too small, for example $k=1$, the analysis does not fully utilize the temporal coherence information of objects in successive frames. If the k value is too large, false positive results will be propagated to many frames. Based on the above-discussed facts, the study experimentally examined the selection of k and found a k value of 4 is suitable in this study.

Algorithm 1 Temporal Coherence Analysis for Recovering False Negative Results.

// N: the number of video frames;
// M_i : the number of objects in frame i;
// O_i : the set of confidently predicted objects in frame i;
// $S_{i,j}$: the prediction score of the jth object in frame i;
// $C_{i,j}$: the center location of the jth object in frame i;
// t_l and t_u: prediction thresholds;
// k: the number of frames that have stored temporal coherence information of predicted objects.

for $i = 1$ *to* N **do**
  for $j = 1$ *to* M_i **do**
    if $S_{i,j} \geq t_u$ **then**
      add object $o_{i,j}$ to the set O_i with its prediction score, $S_{i,j}$, and the center location, $C_{i,j}$
    else if $S_{i,j} \geq t_l$ **then**
      for $q = 1, 2, ...(k-1)$ **do**
        if $\exists\, o_{i-q,j'} \in O_{i-q}$ & $o_{i-q-1,j''} \in O_{i-q-1}$, $\ni\, o_{i,j} = o_{i-q,j'} = o_{i-q-1,j''}$ & $||C_{i,j} - C_{i-q,j'}||_2 \leq q\Delta d$ & $||C_{i,j} - C_{i-q-1,j''}||_2 \leq (q+1)\Delta d$, **then**
          let $S_{i,j} = (S_{i-q,j'} + S_{i-q-1,j''})/2$,
          add object $o_{i,j}$ to the set O_i with its $C_{i,j}$ and updated $S_{i,j}$,
          break
        end if
      end for
    end if
    Eliminate the low score ($< t_u$) object $o_{i,j}$ from the candidate list.
  end for
end for

The range of detection thresholds for the temporal coherence analysis $[t_l, t_u]$ should be appropriately chosen. The upper boundary t_u should be high enough but not extremely high to properly control both types of false results. The lower boundary t_l should be below the upper boundary with a sufficient span to capture most false negative results. Selecting an extremely small lower boundary just increases the workload of temporal coherence analysis, but it has minimal impact on the result due to the control effect of the upper boundary. This study chose $[0.5, 0.9]$ as the threshold range whose appropriateness was verified on small-scale experiments.

The temporal coherence analysis identifies and picks up samples that the current network fails to predict correctly. Therefore, the proposed self-training method effectively learns from its weakness in each round of iterations. Since this temporal coherence is applied in the inference stage, it is a computationally cheap approach for evaluating and sampling new data for annotation.

Algorithm 2 Iteratively Fine-Tuning the Network with S^{3}T.

// l: index of iteration;
// T_l: the training dataset for iteration l;
// V: unlabeled dataset for S^{3}T;
// R_l: the recovered hard dataset from temporal coherence analysis;
// S_l: a subset of R_l, which is sampled based on the sampling method SP(s);
// $S_{l,\alpha}$: a fraction of S_l in the size of α automatically annotated by the trained network;
// $S_{l,(1-\alpha)}$: a fraction of S_l in the size of $1-\alpha$ to be manually annotated by the inspector;
// M_l: the data annotated by the inspector and added to the training dataset in iteration l.

for $l \geq 0$ **do**
  Fine-tune the network with T_l,
  break if the performance meets the requirement.
  obtain R_l through the temporal coherence analysis,
  if $l = 0$ **then**
    sample S_l from the prediction result,
  else
    sample S_l from R_l using the skip sampling method,
  end if
  Split S_l into two mutually exclusive parts,
  manually annotate $S_{l,(1-\alpha)}$ to obtain M_l,
  $T_{l+1} = T_l \cup M_l \cup S_{l,\alpha}$,
  increase α to lower the inspector's workload in data annotation if applicable.
end for

Refining the Network Through Self-Training with Human-in-the-Loop

After transfer learning has initialized the Mask R-CNN for the task of bridge inspection, the network may need to be further refined, for example, by adding additional training data. If the refined network has not reached a satisfying performance, the refining process will continue. To lower the cost of data annotation and, meanwhile, maintain a good quality of the training data, the study chose the S^{3}T method that engages the inspector in continuous refinement of the network. In each iteration, a set of unlabeled data is fed to the trained deep neural network to be labeled automatically. Using the temporal coherence information of predictions, hard samples are collected from this newly created labeled data. A representative subset of the hard samples is identified and added to the training dataset to refine the network. Before this subset is added to the training dataset, a portion of it is manually re-annotated by the inspector to guide the network's learning. This process continues iteratively until the network performance reaches the target. The S^{3}T method with human-in-the-loop is further summarized as the pseudo-code in Algorithm 2.

Let l denote the index of iterations. Figure 4 shows that a small training dataset T_0 is created to transfer the Mask R-CNN. Then, the network is applied to an unlabeled dataset, V. If the performance of the network is not satisfying, a portion of the prediction result is sampled as the additional training data, denoted as S_l. For the initial iteration when $l = 0$, this S_l is taken from all prediction result to guide

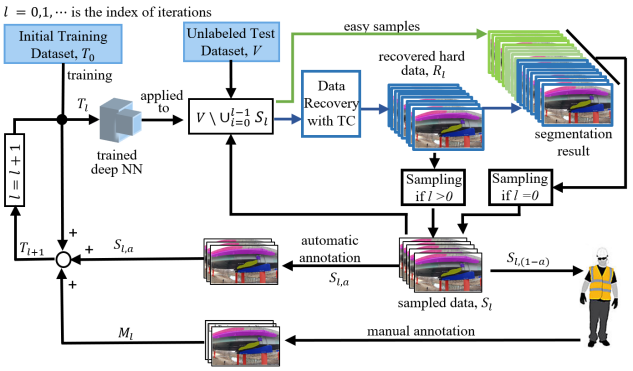


Figure 4. Schematic diagram of the semi-supervised self-training (S^{3}T) for refining the Mask R-CNN model iteratively.

the network. In each iteration, the selected sampled data S_l is eliminated from V for the further assessment of future networks. For the following iterations (i.e., $l > 0$), the S^{3}T algorithm differentiates hard samples from easy samples in V. Easy samples are segmented by the network with relatively high reliability whereas hard samples, R_l, which contain a variety of situations when objects are difficult to detect, are recovered by the developed temporal coherence analysis. The sample S_l (for $l > 0$) is selected only from the recovered hard samples R_l. S_l is divided into two mutually exclusive and collectively exhaustive subsets, $S_{l,\alpha}$ and $S_{l,1-\alpha}$, where α and $1-\alpha$ indicate their sizes in proportion to S_l. $S_{l,\alpha}$ has been automatically annotated by the trained network in testing and directly added to the training dataset. The inspector re-examines the remainder of S_l and corrects false predictions, if any, before adding the inspector-annotated data M_l to the training dataset. That is, at the end of the l^{th} iteration the training dataset is updated per Eq. (1):

$$T_{l+1} = T_l \cup M_l \cup S_{l,\alpha}. \tag{1}$$

The network is re-trained using the updated training dataset and the prediction result is assessed. If the termination criterion has been met, the fine-tuning process is terminated. Otherwise, it continues refining the network. The termination criterion of the iterative process is subject to the user's choice considering the consequences of false positive and false negative results, respectively. This study chose to terminate the iterative process when both precision and recall reach 90% or higher, and f1-score is 92% or higher, at the Intersection over Union (IoU) value 0.5.

This iterative process has two designs: the method for sampling S_l from R_l, and the way of determining the fraction of S_l to be examined by the inspector.

Skip Sampling Method, SP(s). Consecutive frames of a video are similar and, therefore, sampling a portion of frames that are evenly distributed on the timeline would be sufficient for representing the video. This study samples S_l from R_l, for any iteration l, according to a skip sampling strategy $SP(s)$ that samples a frame and then skips s frames. The choice of a value for s needs to consider the inspection platform's speed and the camera speed. The unlabeled test dataset V is a time series of N_v frames. I_{SP} is a $1 \times N_v$ indicator vector of binary variables that define frames to be

sampled according to $SP(s)$; that is,

$$I_{SP}(n) = 1, \tag{2}$$

for $n = 1, 1+(s+1), \ldots, 1+(s+1)\lfloor N_v/(s+1)\rfloor$. $I_{R,l}$ is also a $1 \times N_v$ indicator vector of binary variables that identify the frames recovered by the temporal coherence analysis. The Hadamard product of $I_{R,l}$ and I_{SP} yields the vector $I_{S,l}$,

$$I_{S,l} = I_{R,l} \circ I_{SP}, \tag{3}$$

which identifies the frames to be sampled from R_l according to $SP(s)$ for forming S_l.

Regulating the Amount of Human Guidance in S^{3}T. A fraction of the dataset S_l from any iteration is automatically annotated by the trained network. The initial performance of the neural network is not high and data mislabeled by the network are present in S_l. Through examining a fraction of S_l and correcting mislabeled data, the experienced inspector guides the network to quickly learn new features. $S_{l,\alpha}$ is the fraction of S_l which is added to the training dataset without further human annotation. The inspector's guidance can be gradually reduced as the network starts to learn well by itself and provide improved prediction. Therefore, the fraction of automatically annotated data $S_{l,\alpha}$ can be gradually increased over iterations. Choosing the initial value of α is also critical as it regulates the amount of mislabeled data that may enter the training dataset when the model performance is well below the target performance. Determining an optimal selection of α for the S^{3}T method is a research problem but going beyond the scope of this paper. The paper illustrates the impact of choosing α in Table 5.

Implementation and Result Discussion

This section illustrates the implementation and evaluation of the proposed method to create the assistive intelligence model for processing the bridge inspection video data. Findings from this study are discussed.

The Implementation Detail

The Data. BIRDS[48], an aerial inspection platform developed by the INSPIRE University Transportation Center, was used to capture videos of bridges in inspection. The average speed of BIRDS is 20 miles per hour (mph). The frame rate of the camera is 30 frames per second (fps) and the image resolution is 3,840 × 2,160 in pixel. A dataset D, which is an inspection video of 4,440 images, was used to develop and evaluate the assistive intelligence model. The initial training dataset T_0 contains 40 images, with 18 images from D and 22 images from the inspection of other bridges. Choosing some images of other bridges adds helpful data variation to the initial training dataset. In total, the initial training dataset contains 482 objects with class labels, which are from 10 different classes of bridge elements interested to inspect. The 10 object classes are barrier, slab, pier, pier cap, diaphragm, joint, bearing, pier wall, bracket, and rivet, illustrated in Figure 5. This study used the image annotation tool VGG Image annotator[49] to annotate labels of the objects and give pixel-level coordinates to those objects. An unannotated dataset V that comprises 670 images from the dataset D was particularly created for implementing the S^{3}T method. V contains 5,916 objects from the 10 classes. A test dataset T_s has been created to evaluate the model performance from each training iteration. This dataset has 212 images with 1,872 objects.

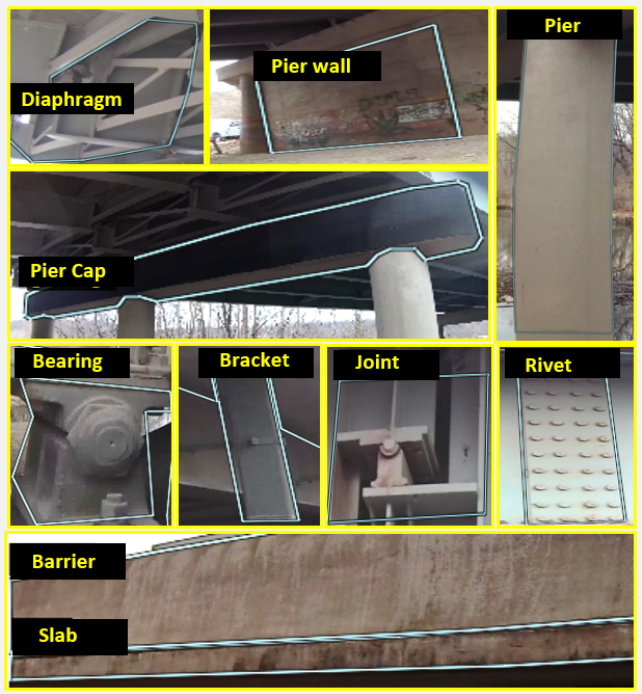


Figure 5. Sample images with corresponding pixel-level object polygon with labels.

Initial Adaption. The proposed method was implemented by extending an existing implementation of Mask R-CNN by Matterport Inc.[50]. Training and testing were performed using two Nvidia Tesla V100 GPUs with 32GB of memory. The pre-trained ResNet-50 feature extractor was fine-tuned using the initial training dataset T_0. Different data augmentation techniques have been applied during the training stage to improve the model's ability to generalize at various applicability where input data distortion is present. Those include horizontal flip, rotation, translation, color distortion, and random noise. The network head and the mask head (see Figure 2) were trained for 30 epochs while keeping all the parameters of the previous layers fixed. Each epoch consists of 100 training iterations. Stochastic gradient descent was used as the optimizer and the momentum was 0.9. The learning rate of 0.001 and a batch size of 4 were used in this training process that took about 21 minutes to complete. According to Algorithm 2, after the Mask R-CNN is transferred to have an initial adaption to the task of bridge inspection, V is annotated by this network. Considering the performance of the initially adapted network, 8 images, which are about 1% of the images in V, are selected and re-annotated by the inspector and added to the initial training dataset T_0, becoming T_1, the training dataset for the next iteration. These 8 images are excluded from V for further iterations. Letting the inspector to check a small amount of the prediction result of the initially adapted network in accordance with the performance is a practical approach to controlling the quality of the training dataset.

Table 1. Data sizes (# images) in transfer learning (TL) and semi-supervised self-training (S^{3}T).

l : index of iterations	TL	S^{3}T		
	0	1	2	3
T_l: training dataset (Eq. 1)	40	48	85	118
R_l: recovered hard data samples	-	113	79	50
S_l: a sampled subset of R_l	-	37	33	
α: % of S_l for automatic annotation	-	70%	80%	
$S_{l,\alpha}$: automatically annotated data	-	26	26	
M_l: manually annotated data	8	11	7	

Inference and Iterative Refining. Table 1 summarizes the iterative training process for fine-tuning the deep neural network using the S^{3}T method with the inspector's guidance. Refining the network for the 1st iteration of the S^{3}T method was initiated with the last epoch of the previous iteration (i.e., the transfer learning) and continued for 20 more epochs. Then, the remainder of the dataset V, $V \setminus S_0$, is annotated by the refined network. Temporal coherence analysis is applied to $V \setminus S_0$, which contain objects with prediction scores between 0.5 and 0.9 to recover false negative results. This study considered 0.5 as the lower boundary of prediction threshold t_l and 0.9 as the upper boundary t_u, which were found to minimize the volume of false negative results based on numerical experiments. The temporal coherence analysis in the 1st iteration recovered 113 frames, and 37 frames of these were sampled according to SP(2), the sampling strategy considered by this study. In this study, α was 70% in the 1st iteration, which means the inspector re-annotated 30% of S_1 before adding them to the training dataset. In the 2nd iteration, the network was refined using the updated training dataset T_2 and then it was used to evaluate $V \setminus (S_0 \cup S_1)$. Temporal coherence analysis recovered 79 images and 33 images were sampled and added to the training dataset. α was increased to 80% of S_2 and the inspector annotated only 7 images out of the 33 before adding them to the training dataset. The iterative process was terminated after the 3rd iteration of training when the target performance is achieved.

Quantitative Results

In this study, experiments were conducted to evaluate the merits of the proposed method, the job efficiency of the developed assistive intelligence model, and its generalization capacity.

Object Detection Results. To evaluate the performance of object detection with the developed deep neural network, three standard evaluation matrices were used in this study:

- precision: it counts the number of correct predictions out of the total number of predictions;
- recall: it counts the number of correct predictions out of total number of ground-truth objects;
- f1-Score: it is the harmonic mean of precision and recall.

This study used the Intersection over Union (IoU) to determine whether a predicted object can be considered as a correct prediction. The IoU is the intersection between the predicted bounding box and the ground truth bounding box over the union of them. The ability of the network to correctly detect objects was evaluated on a range of IoU threshold values from 0.1 to 0.9 at a step of 0.1. The precision, recall, and f1-score from evaluating the test dataset T_s are summarized in Table 2. From the table, it can be observed that, given an IoU threshold value in the range of [0.1, 0.5], the Mask R-CNN that was initially transferred in for the bridge of inspection achieved the precision from 80.3% to 87.5%, the recall from 74.4% to 81.0%, and f1-score from 77.2% to 84.1%. The performance indicates the transferred network demonstrated some adaptability to the new task, but the amount of false negative detection is non-negligible. The performance of the network has not reached a satisfying level.

Table 2. Performance (%) of transfer learning (TL) and semi-supervised self-training (S^{3}T) in iterations.

IoU	Iteration, l	TL	S^{3}T		
		0	1	2	3
0.1	Precision	87.5	86.3	94.0	93.45
	Recall	81.0	95.4	93.5	95.30
	F1-Score	84.1	90.6	93.8	94.4
0.2	Precision	87.1	85.7	93.9	93.5
	Recall	80.7	94.8	93.3	95.3
	F1-Score	83.4	90.0	93.6	94.4
0.3	Precision	86.8	85.2	93.7	93.4
	Recall	80.3	94.2	93.2	95.2
	F1-Score	83.4	89.5	93.4	94.3
0.4	Precision	84.6	84.2	93.2	93.1
	Recall	78.3	93.1	92.7	94.9
	F1-Score	81.3	88.5	93.0	94.0
0.5	**Precision**	80.3	81.7	90.7	91.8
	Recall	74.4	90.3	90.1	93.6
	F1-Score	77.2	85.8	90.4	92.7
0.6	Precision	75.9	77.4	85.7	88.5
	Recall	70.2	85.6	85.1	90.2
	F1-Score	73.0	81.3	85.4	89.3
0.7	Precision	65.4	66.6	74.6	78.1
	Recall	60.5	73.6	74.2	79.6
	F1-Score	60.5	73.6	74.2	79.6
0.8	Precision	43.7	43.8	50.1	49.0
	Recall	40.5	48.5	49.8	49.9
	F1-Score	42.1	46.0	49.9	49.5
0.9	Precision	6.3	3.0	6.7	4.6
	Recall	5.9	3.3	6.7	4.7
	F1-Score	6.1	3.1	6.7	4.7

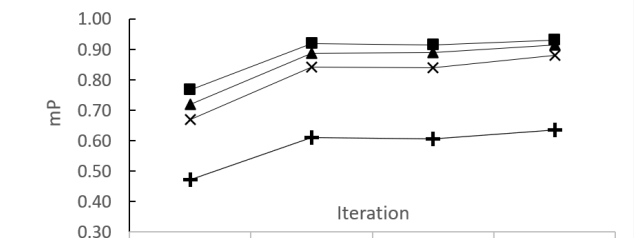
Therefore, the network was iteratively refined using the proposed S^{3}T method with human-in-the-loop to seek further improvement. After being re-trained in the 1st iteration, the recall was effectively increased by 15%, approximately. For example, when the IoU threshold value is 0.5, the precision increases from 80.3% to 81.7%, and recall becomes 90.3% from 74.4%, yielding a 85.8% f1-score after the 1st iteration. The changes indicate that M_0, the additional small set of manually annotated hard samples added to the training dataset, effectively improves the ability to correctly detect more objects. The performance of the network has met the requirement after being refined for additional two iterations, reaching 91.8% precision, 93.6% recall, and 92.7% f1-score at the IoU threshold value 0.5. The S^{3}T method has effectively brought the performance of the network to a satisfying level. As the IoU threshold value decreases gradually from 0.5 to 0.1 at a step size of 0.1, the evaluation becomes less conservative.

Consequently, fewer false negative detections are rendered by the network but maybe more false positive detections. On the other hand, selecting a higher IoU threshold value makes the evaluation more conservative. As it increases gradually from 0.5 to 0.9 at a step size of 0.1, the f1-score is diminishing, signifying the reduction of both precision and recall values. In this application setting, false positive detections are less concerned than false negative detections. This is because the inspector will retrieve and analyze frames that contain detected and segmented objects that s/he wants to inspect. Therefore, false positive detections can be found and eliminated by the inspector. But, false negatives are more critical because inspectors cannot overlook any potential damages. From the analysis above, it can be inferred that this S^{3}T method is very applicable to the development of the proposed assistive intelligence model for detecting bridge elements from inspection videos.

Another important observation from the table is the relationship between the IoU threshold value and the recall value during the iterative process of network fine-tuning. A variation of a recall value within a range of 6.6% has been observed after the initial adaption through transfer learning when the IoU threshold value increases from 0.1 to 0.5. However, this variation reduces in each successive iteration. For example, after the third iteration, this variation reduces to 1.6%. Precision and recall values at any of the IoU threshold values increase over iterations and reach the maximum after the 3rd iteration. For example, at the IoU threshold value 0.5, f1 score increases about 8.6%, 4.6%, and 2.3%, respectively, from their previous iteration. This means the network learns new features from each iteration and gradually moves toward the learning limit. The improvement rate is diminishing during the iterative process. When the IoU threshold value continues increasing from 0.5 and onward, the recall value in any iteration drops rapidly and has been less than 10% when the IoU threshold value is 0.9. Moreover, the increasing trend of the recall value over iterations slows down quickly at the IoU threshold value 0.6. When the IoU threshold value is greater than 0.6, the increasing trend of recall over iterations is rapidly flatted out and becomes a decreasing trend. The reason for this sharp decrease of recall value along with the increase of the IoU threshold value is that the network considers a detected object as a true positive detection only if the overlap between the ground truth and the bounding box of the detected object is very high, which increases the amount of false negative detections and decreases the amount of true positive detections.

Instance Segmentation Results. The study also evaluated the quality of the proposed assistive intelligence model in segmenting bridge elements from inspection videos. The mask Intersection over Union (IoU) is a measure of segmentation quality, which is the ratio of the overlap between the predicted segmentation mask and the ground truth mask to the union of these two masks. The predicted segmentation mask is considered as a true positive prediction if the mask IoU value is no less than a pre-specified threshold. The higher the threshold value is selected for the evaluation, the stricter the evaluation becomes. Accordingly, The precision for each class can be calculated. After that, the average of the class-level precision values, named mean precision and denoted as mP, is determined.

Figure 6 shows the curve of mP value during the iterative process for fine-tuning the network at four levels of mask IoU threshold value, wherein the x-axis represents the number of iterations and the y-axis represents the mP value. The plot shows the mP curve at the mask IoU threshold value 0.4 is an upwarding curve on the top of other curves. The mP value at the end of the iterative process reaches 93%. When the mask IoU threshold value increases to 0.5, the curve just drops slightly and the shape of the curve has no change. The mP value at the end of the iterative process reaches 92%. However, with a larger mask IoU threshold value such as 0.75, the mP curve clearly drops to a lower position. This is because the amount of true positive results at a larger mask IoU threshold value is low although the total number of correctly segmented objects increases over iterations.



	0	1	2	3
mP@40	0.77	0.92	0.92	0.93
mP@50	0.72	0.89	0.89	0.92
mP@60	0.67	0.84	0.84	0.88
mP@75	0.47	0.61	0.61	0.64

Figure 6. The mean Precision (mP) over iterations at different mask IoU threshold values.

Efficiency of Transfer Learning. To demonstrate the high cost-effectiveness of transfer learning for the initial adaption, this study trained a Mask R-CNN from the scratch using 144 annotated images. Results of the comparison are summarized in Table 3. The first 600 epochs for training the network from scratch took 13.2 hours, and the network performance (32.3% precision, 18.3% recall, and 23.4% f1 with an IoU threshold value 0.5) is well below the target performance when it was tested on the dataset T_s. The experiment clearly demonstrates the network requires a huge number of training samples to be trained from the scratch, which is infeasible for developing the desired model of inspection video data analysis due to the scarcity of labeled data. The proposed transfer learning used only 40 annotated images as the training dataset and took only 20 minutes to transfer the capability of an existing Mask R-CNN in multiclass object detection and segmentation to the new task with bridge elements. The performance of the transferred Mask R-CNN has a much better result (80.3% precision, 74.4% recall, and 77.2% f1 at the IoU 0.5). The comparison summarized in Table 3 demonstrates that transfer learning reduced the training time by at least 95% and has improved the performance of the network tremendously.

Comparison with a State-of-the-Art Method. This study compared the proposed approach (i.e., transfer learning plus S^{3}T with human-in-the-loop) to the Mask R-CNN adapted

Table 3. Cost-effectiveness of transfer learning in comparison with training from the scratch.

	Training time (hrs)	Precision (%)	Recall (%)	F1 (%)
Training from scratch	13.2	32.3	18.3	23.4
Transfer learning	0.33	80.3	74.4	77.2

with transfer learning only and the traditional self-learning, from the perspectives of annotation time, training time, inference speed, and accuracy. To show the reliance of the performance of the transferred Mask R-CNN on the volume of training dataset, this study measured the performance of the Mask R-CNN after independently transferred with four random training datasets: 0.5% (22 images) , 1% (44 images), 5% (220 images) and 10% (440 images) of images in Dataset D. Results from the comparison are summarized in Table 4. It is observed that the transfer learning by itself can improve the performance of the network, but the improvement is at a rapidly increasing cost of annotation time. Transferring the Mask R-CNN with 440 annotated images took 1430 minutes (i.e., 23.8 hours) for data annotation and 66 minutes for training. This network achieves 89.7% precision, 92.3% recall, and 91.0% f1 score, close to the performance of the proposed approach in this paper. The proposed approach reduces the annotation time by 85% with a comparable training time (only 6 minutes longer), and it achieves a better performance (91.8% precision, 93.6% recall, and 92.7% f1 score). Self-training does not use the inspector guidance in the iterative re-training process, thus saving about one hour of annotation time compared to the S^{3}T with human-in-the-loop. But the performance is not satisfying. The inference speed of all the models is 0.55 seconds per frame. This comparative study demonstrated that the S^3T method with human-in-the-loop is more cost-effective compared to directly transferring the Mask R-CNN. It also significantly improves the model performance compared to traditional self-training, due to the engagement of experienced inspectors in the iterative re-training process. The impact of human-in-the-loop on self-training is further examined.

The Impact of Human-in-the-Loop on Self-Training. The S^{3}T method that keeps human-in-the-loop is a combination of self-training and active learning. The portion of additional training data re-annotated by experienced inspectors in each iteration of the self-training process may impact the efficiency of model development and the performance of the resulting final model. This study used four experiments to illustrate the impact of human-in-the-loop on self-training, which are summarized in Table 5. The four experiments all began with the same initial model whose performance is 80.3% precision, 74.4% recall, and 77.2% f1-score. In all the experiments the model is trained for three iterations. Inspectors annotated eight images to re-train the initial model for the first iteration. With this iteration, the performance is increased to 81.7% precision, 90.3% recall, and 85.8% f1-score. After that, the four experiments differ in the ratio of model-annotated additional training data (α). If an experiment yields satisfying model performance (i.e., both recall and precision are at least 90%, and f1-score is at least 92% at the IoU threshold value 0.5) within three iterations, the final performance is highlighted as bold.

Experiment 1 is our approach in Tables 2 and 4. Compared to experiment 1, experiment 2 let the inspector re-label more data for the second and third iterations of training. It not only provided a satisfying performance at the end of iteration 3 but increased the f1-score with a small margin over that from experiment 1. Compared to experiment 1, experiment 3 used less inspector-annotated additional training data for the third iteration. Although it does not achieve the target performance at the end of the third iteration, f1-score is just below the target by 0.1% a tiny margin. That is, the target performance is likely to be reached in the next iteration of training. Unlike experiments 1∼3 that are all self-training with human-in-the-loop, experiment 4 is the traditional self-training without human-in-the-loop. That is, the model teaches itself by adding additional data to the training dataset iteratively. The added new data are those with the most confident label prediction by the model. To make it a fair comparison, experiment 4 used the same amount of additional training data as experiment 1 in each iteration except that the additional data for iterations 2 and 3 are all model-annotated. By the end of the third iteration, the model performance is well below the target performance. The drop of the model performance from the first iteration to the second iteration, and the slow increase of the performance from the second iteration to the third iteration, indicate including the inspector's guidance in self-training is critical.

The proposed S^{3}T method with human-in-the-loop (in experiments 1∼3) has tremendously improved the efficiency of model development as it is compared to the supervised learning method (Mask R-CNNs in Table 4). Its performance improvement has a large margin over the traditional self-training (in experiment 4). Further optimizing the engagement of inspectors in self-training has just incremental improvement against the heuristic strategy of this paper.

Job Efficiency of the Assistive Intelligence Model. A full image usually contains multiple elements of the bridge. An inspector needs to search and find all the elements, segment each identified element by marking its boundary, and provide the object name. On average, each image in this study contains 15 objects of different sizes and shapes. To provide better quality, a polygon rather than a rectangle is preferred for segmenting identified bridge elements. Drawing a tight polygon on a single object may require defining 15∼30 points on the image. On average, it took around 3.25 minutes to detect and manually segment bridge elements in a full image in this study. Not to mention that issues related to human factors, such as the fatigue developed from repeatedly working on high cognitive tasks, further lengthen the time required for manually analyzing the big video data collected from bridge inspection. The approach that this paper proposes requires inspectors to analyze a small amount of data manually, helping the network achieve satisfying performance in object detection and segmentation. The automated data processing relieves inspectors from the time-consuming labor work. The developed assistive intelligence model can finish the same job with very high

Table 4. Comparison of the proposed approach to directly transferring a Mask R-CNN with various volumes of training dataset.

Method	No. of manually annotated images	Annotation time (min)	Training time (min)	Inference speed (sec/frame)	Precision (%)	Recall (%)	F1-score (%)
Mask R-CNN	22	72	18	0.55	68.0	68.4	68.2
Mask R-CNN	44	143	20	0.55	82.0	79.0	80.5
Mask R-CNN	220	715	33	0.55	85.8	91.8	88.7
Mask R-CNN	440	1,430	66	0.55	89.7	92.3	91.0
Self-training	48	156	72	0.55	88.9	76.7	82.4
Our approach	**66**	**215**	**72**	**0.55**	**91.8**	**93.6**	**92.7**

Table 5. Variants of inspector engagement in self-training

	iterations	0	1	2	3
1	precision (%)	80.3	81.7	90.1	**91.8**
	recall (%)	74.4	90.3	90.7	**93.6**
	f1-score (%)	77.2	85.8	90.4	**92.7**
	$S_{l,\alpha}$ (frame)	0	26	26	
	M_l (frame)	8	11	7	
	α_l (%)	0	70	80	
2	precision (%)	80.3	81.7	91.8	**94.5**
	recall (%)	74.4	90.3	84.7	**91.5**
	f1-score (%)	77.2	85.8	88.1	**93.0**
	$S_{l,\alpha}$ (frame)	0	22	26	
	M_l (frame)	8	15	11	
	α_l (%)	0	60	70	
3	precision (%)	80.3	81.7	90.1	91.7
	recall (%)	74.4	90.3	90.7	92.1
	f1-score (%)	77.2	85.8	90.4	91.9
	$S_{l,\alpha}$ (frame)	0	26	28	
	M_l (frame)	8	11	3	
	α_l (%)	0	70	90	
4	precision (%)	80.3	81.7	88.3	88.9
	recall (%)	74.4	90.3	75.7	76.7
	f1-score (%)	77.2	85.8	81.5	82.4
	$S_{l,\alpha}$ (frame)	0	37	33	
	M_l (frame)	8	0	0	
	α_l (%)	0	100	100	

accuracy but with only 0.55 seconds per image, which is 350 times faster than the manual approach. The impact of the improved work efficiency is tremendous because a real-world task usually requires analyzing hundreds of thousands of images.

In the real-world implementation, the developed assistive intelligence model will detect and segment bridge elements from the inspection video data. Then, the inspector retrieves the elements of interests from the large pool of video frames and evaluates damages or other defects associated with them. The developed model assists inspectors in that, it reduces the human effort in searching and finding the needed data so that inspectors can focus on knowledge-intensive tasks. Moreover, by removing the burden of browsing hours of videos to look for bridge elements for evaluation, inspectors are less likely to work in a state of fatigue or lacking focus.

Generalization Capability of the Proposed Method. To assess if the developed network is applicable to other bridges, the study used it to detect and segment the same ten classes of bridge elements of another two bridges, named bridges A and B. Correspondingly, the bridge that has had a network developed for is named bridge C. When the network built for bridge C was applied to bridges A and B, the performance of it is comparable to that of the initial network for bridge C. The trained network achieves 76.8% precision, 73.0% recall, and 74.8% f1-score for bridge A; and for bridge B it accomplishes 61.2% precision, 60.0% recall, and 60.6% f1-score. Bridge A is more similar to bridge C than bridge B. Therefore, the network trained for bridge C performs better in analyzing the inspection data of bridge A than bridge B. Therefore, the network for bridge C has a certain degree of generalization, and it is good enough to serve as the initial network for other bridges. Further implementing the S^{3}T method developed in this paper will adapt the assistive intelligence model developed for bridge C to bridges A and B, respectively, to achieve the target performance on these bridges.

Qualitative Results

The developed assistive intelligence model was tested on the inspection dataset D. Some qualitative examples selected from the testing result are illustrated below.

An Illustrative Example of Temporal Coherence Analysis. Figure 7 illustrates an example wherein the temporal coherence analysis improves the model performance by eliminating false negative results. In the first row of Figure 7, the single-image based network correctly predicted the diaphragm in the 1st and the 4th frames, however, failed to detect it in the 2nd and the 3rd frames. The second row is the result after applying the temporal coherence analysis, which shows that the diaphragm was correctly segmented in all of the four frames. Note that false negative results are more severe than false positive results in the task of bridge inspection. Because false positives rendered by the deep network can be re-checked by the inspectors but false negatives ignored by the network will not have such an opportunity. Therefore, effectively reducing false negative results is particularly more important for the bridge inspection.

Representative Examples of Successful Detection and Segmentation. Figure 8 (a) illustrates some representative examples of successful segmentation of bridge elements by the developed network. The first column of Figure 8 (a) is an example that a partial joint of different scales in three images is detected and segmented correctly from all images. The second column illustrates the capability of detecting and segmenting a rivet from various views. The network successfully detects and segments the rivet in a low light condition, as the second figure in column two illustrates. The third column comprises successful examples of segmenting the rivet at a wide range of scale variations. The fourth column shows that the developed network is

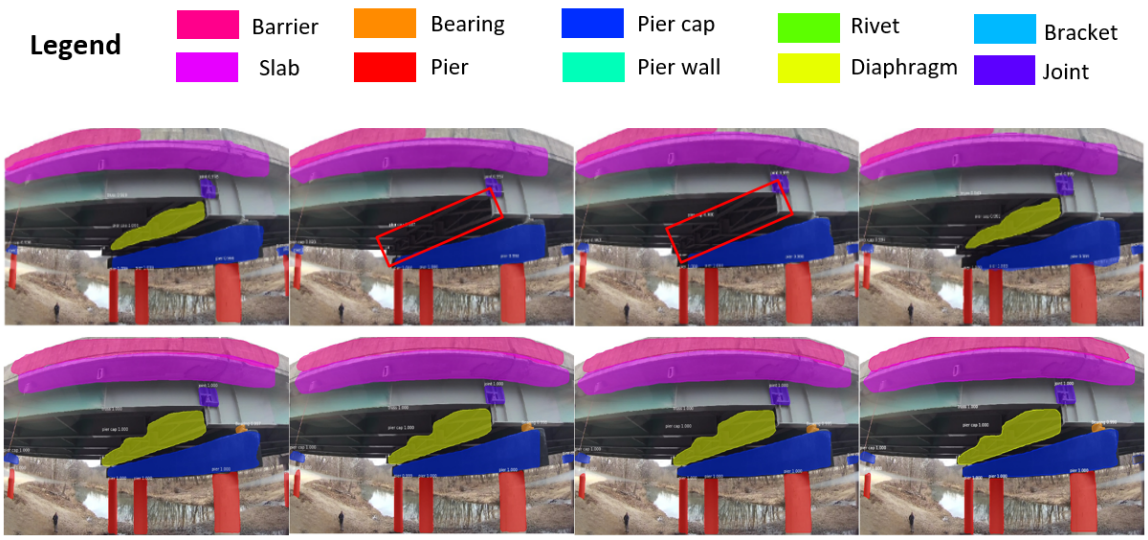


Figure 7. An example of temporal coherence analysis.

successful in detecting and segmenting multiple objects at various distances in complex scenes.

Representative Examples of False Negative Detection The red bounding boxes in Figure 8 (b) represent false negative results by the developed network. The diaphragm in the top frame and the pier cap in the middle frame are difficult to recognize because of the dark illumination level. In the bottom frame, the developed network fails to detect the barrier due to the high exposure. The rivet in the bottom frame is not detected because of its similar appearance with the background. External illumination sources on inspection platforms and image contrast enhancement techniques are potential solutions to overcome these challenges.

Conclusions

This paper presents a method to develop an assistive intelligence model to support bridge inspectors in segmenting multiclass bridge elements from big complex video data collected by aerial inspection platforms. With a small initial training dataset annotated by inspectors, a Mask R-CNN pre-trained on a large public dataset was transferred to the new task of bridge inspection. Then, the temporal coherence analysis was used to recover false negative results and thus identify the weakness of the current network to improve, which adds a nearly negligible additional computation load during the inference compared to other methods based on the motion guidance. A semi-supervised self-training algorithm was developed to engage the inspector in refining the network iteratively. The domain knowledge of the inspector quickly brought the network's performance to a satisfying level.

Assessment results of the developed assistive intelligence model showed that the proposed approach to the model development uses a small amount of time and guidance from bridge inspectors to achieve a high performance in segmenting multiclass structural elements from the big complex inspection videos. For example, the developed model has achieved around 94% of precision, 92% of recall, and 92% mAP when the IoU threshold value is 0.5. The study revealed that having sufficient guidance from experienced bridge inspectors, particularly in early iterations of the semi-supervised self-training for refining the network is critical for maintaining the quality of the training dataset. The amount of human annotation can be gradually reduced as the network becomes more reliable in performing its tasks.

The paper has identified rooms for improvement. Adapting the assistive intelligence model to bridges with additional structural elements is the next step to extend this paper. One important future work is to improve the inference speed. While the developed model is able to achieve a high performance with a small amount of human hours and the computation time for training the network, improving the testing speed to have the real time inference capability is highly desired. Moreover, contextual information and the spatial correlation among objects could be utilized to further improve the segmentation accuracy. Another Future work will evaluate the change in cognitive load and other psychological states of inspectors assisted by the assistive intelligence in bridge inspection. The evaluation involves using biometric sensors such as eye movement trackers and electroencephalogram (EEG) to detect heavy cognitive load, fatigue, and lost of focus of inspectors in their tasks.

Acknowledgements

Research presented in this paper is partially supported by the U.S. Department of Transportation, Office of the Assistant Secretary for Research and Technology (USDOT/OST-R) under Grant No. 69A3551747126 through INSPIRE University Transportation Center (http://inspire-utc.mst.edu). The views, opinions, findings and conclusions reflected in this publication are solely those of the authors and do not represent the official policy or position of the USDOT/OST-R, or any State or other entity.

Appendix

Figure 9 below describes an object in the 3D camera reference frame and the 2D image pixel reference frame. At a certain time, the coordinates of the object in the camera reference frame is $(x, y, z)^T$, and $(x_m, y_m)^T$ in the image

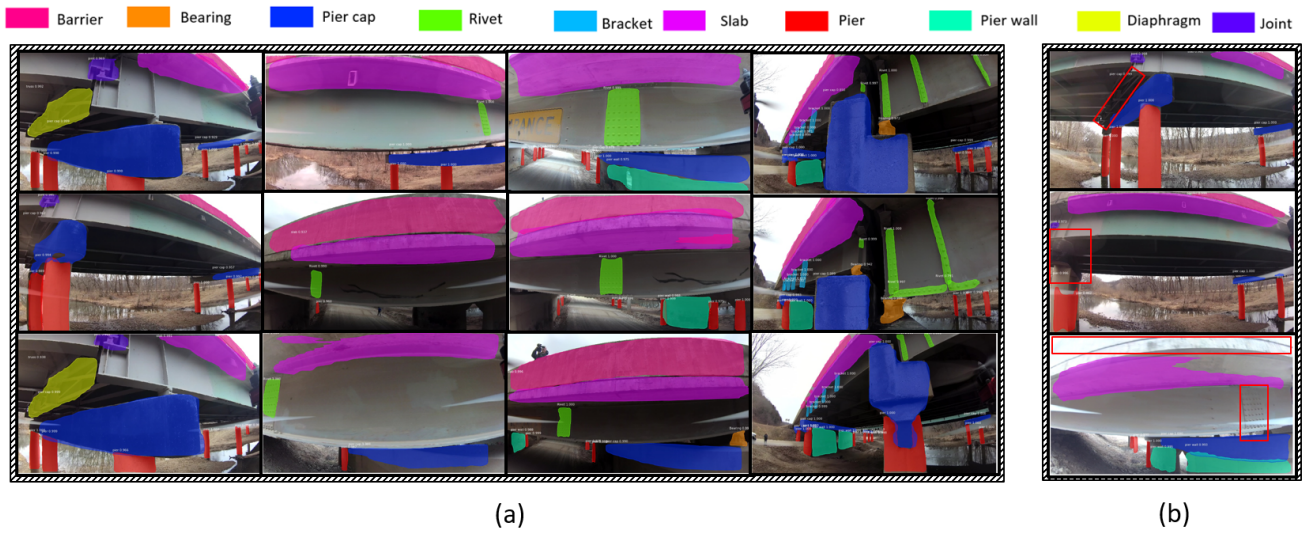


Figure 8. Examples of (a) successful detection and segmentation, and (b) false negative detection.

pixel reference frame. The relationship of these two sets of coordinates of the object is determined as:

$$\begin{bmatrix} x_m \\ y_m \end{bmatrix} = \begin{bmatrix} o_x \\ o_y \end{bmatrix} - \frac{f}{s_m} \begin{bmatrix} x/z \\ y/z \end{bmatrix} \tag{4}$$

where $(o_x, o_y)^T$ represent the coordinates of the principal point (in pixel) of images, f is the focal length of camera, and s_m is the size of pixel (millimeter per pixel and it is assumed to be the same on both X_m and Y_m axes in this study). $(o_x, o_y)^T$, f, and s_m are intrinsic camera parameters.

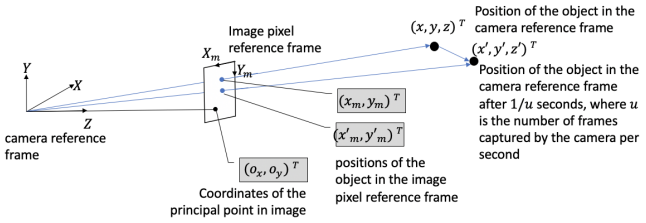


Figure 9. Intrinsic camera parameters define the relationship between the camera reference frame and the image pixel reference frame.

In this study, the object is static, and the camera is moving, in the world reference frame. But in the camera reference frame, the object is moving. Imaging that the object moves from $(x, y, z)^T$ to $(x', y', z')^T$ in $1/u$ seconds in the camera reference frame, where u is the speed of the camera in capturing images (i.e., how many images are taken per second). Accordingly, the object moves from $(x_m, y_m)^T$ to $(x'_m, y'_m)^T$ in the next frame of image. $(x', y', z')^T$ must be within an L2 ball of radius ϵ centered at $(x, y, z)^T$. The radius of the ball is determined by the maximum linear and rotational speeds of the camera. Therefore, the displacement of the object from one frame of image to the next frame is:

$$\begin{aligned} \Delta r &= \sqrt{(x'_m - x_m)^2 + (y'_m - y_m)^2} \\ &= \frac{f}{s_m} \sqrt{\left(\frac{x'}{z'} - \frac{x}{z}\right)^2 + \left(\frac{y'}{z'} - \frac{y}{z}\right)^2} \\ &\leq \frac{f}{s_m} \frac{\epsilon}{\min(|z|, |z'|)} = \Delta d \end{aligned} \tag{5}$$

Eq. 5 shows the maximum displacement of an object (Δd) from one frame to the next frame of image is proportional to the focal length (f) and the radius of the L2 ball that defines the boundary of the object's relative motion rate in the camera reference frame (ϵ). Δd is inversely proportional to the size of pixel s_m and the minimum distance of the camera to the object along the optical axis when the two successive frames are captured ($\min(|z|, |z'|)$). Eq. 5 indicates Δd is affected by intrinsic camera parameters f and s_m. Extrinsic parameters affect Δd too because the term $\epsilon / \min(|z|, |z'|)$ is determined by the position and motion rate of the camera in the world reference frame.

References

1. ASCE. 2017 infrastructure report card. Technical report, American Society of Civil Engineers, 2017. URL https://www.infrastructurereportcard.org/wp-content/uploads/2016/10/2017-Infrastructure-Report-Card.pdf.
2. Henderson G. Highway bridge inspections, 2017. URL https://www.transportation.gov/content/highway-bridge-inspections.
3. AASHTO. *Manual for Bridge Element Inspection*, 2 ed, 2019.
4. Jáuregui DV and White KR. Implementation of virtual reality in routine bridge inspection. *Transportation Research Record* 2003; 1827(1): 29–35.
5. Weatherington D. Unmanned aerial vehicles roadmap: 2002-2027. *Office of the Secretary of Defense* 2002; .
6. FHWA. National bridge inspection standards regulations (NBIS). *Federal Register* 2004; 69(239): 74419–74439.

7. Zhu Z, German S and Brilakis I. Detection of large-scale concrete columns for automated bridge inspection. *Automation in Construction* 2010; 19(8): 1047–1055.
8. Roland T and Harlow C. Automated visual inspection: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 1982; PAMI-4(6).
9. Snyder CR. Selection, inspection, and naming in visual search. *Journal of Experimental Psychology* 1972; 92(3): 428–431.
10. Yeum CM and Dyke SJ. Vision-based automated crack detection for bridge inspection. *Computer-Aided Civil and Infrastructure Engineering* 2015; 30(10): 759–770.
11. Xu Y, Bao Y, Chen J et al. Surface fatigue crack identification in steel box girder of bridges by a deep fusion convolutional neural network based on consumer-grade camera images. *Structural Health Monitoring* 2019; 18(3): 653–674.
12. Hoskere V, Narazaki Y, Hoang TA et al. Madnet: multi-task semantic segmentation of multiple types of structural materials and damage in images of civil infrastructure. *Journal of Civil Structural Health Monitoring* 2020; 10: 757–773.
13. Hu F, Zhao J, Huang Y et al. Structure-aware 3d reconstruction for cable-stayed bridges: A learning-based method. *Computer-Aided Civil and Infrastructure Engineering* 2021; 36(1): 89–108.
14. Dong CZ and Catbas FN. A review of computer vision–based structural health monitoring at local and global levels. *Structural Health Monitoring* 2020; : 1475921720935585.
15. Bao Y, Tang Z, Li H et al. Computer vision and deep learning–based data anomaly detection method for structural health monitoring. *Structural Health Monitoring* 2019; 18(2): 401–421.
16. Deng J, Lu Y and Lee VCS. Imaging-based crack detection on concrete surfaces using you only look once network. *Structural Health Monitoring* 2020; : 1475921720938486.
17. Narazaki Y, Hoskere V, Hoang TA et al. Automated vision-based bridge component extraction using multiscale convolutional neural networks, 2018. ArXiv preprint arXiv:1805.06042.
18. Narazaki Y, Hoskere V, Hoang TA et al. Vision-based automated bridge component recognition with high-level scene consistency. *Computer-Aided Civil and Infrastructure Engineering* 2020; 35(5): 465–482.
19. Yeum CM, Choi J and Dyke SJ. Automated region-of-interest localization and classification for vision-based visual assessment of civil infrastructure. *Structural Health Monitoring* 2019; 18(3): 675–689.
20. Kang K, Li H, Yan J et al. T-CNN: Tubelets with convolutional neural networks for object detection from videos. *IEEE Transactions on Circuits and Systems for Video Technology* 2018; 28(10): 2896–2907.
21. Zhu X, Xiong Y, Dai J et al. Deep feature flow for video recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. pp. 2349–2358.
22. Zhu X, Wang Y, Dai J et al. Flow-guided feature aggregation for video object detection. In *Proceedings of the IEEE International Conference on Computer Vision*. pp. 408–417.
23. Han W, Khorrami P, Paine TL et al. Seq-NMS for video object detection. *arXiv preprint arXiv:160208465* 2016; .
24. Lin TY, Maire M, Belongie S et al. Microsoft COCO: Common objects in context. In *European Conference on Computer Vision*. Springer, pp. 740–755.
25. Girshick R, Donahue J, Darrell T et al. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern recognition*. pp. 580–587.
26. Uijlings JR, Van De Sande KE, Gevers T et al. Selective search for object recognition. *International Journal of Computer Vision* 2013; 104(2): 154–171.
27. Girshick R. Fast R-CNN. In *Proceedings of the IEEE International Conference on Computer Vision*. pp. 1440–1448.
28. He K, Gkioxari G, Dollár P et al. Mask R-CNN. In *Proceedings of the IEEE International Conference on Computer Vision*. pp. 2961–2969.
29. Zhang C, Chang C and Jamshidi M. Bridge damage detection using single-stage detector and field inspection images. *arXiv preprint arXiv:181210590* 2018; .
30. Gao Y and Mosalam KM. Deep transfer learning for image-based structural damage recognition. *Computer-Aided Civil and Infrastructure Engineering* 2018; 33(9): 748–768.
31. Zhang C and Chang CC. Surface damage detection for concrete bridges using single-stage convolutional neural networks. In *Health Monitoring of Structural and Biological Systems XIII*, volume 10972. International Society for Optics and Photonics, p. 109722E.
32. Gopalakrishnan K, Gholami H, Vidyadharan A et al. Crack damage detection in unmanned aerial vehicle images of civil infrastructure using pre-trained deep learning model. *Int J Traffic Transp Eng* 2018; 8: 1–14.
33. Papandreou G, Chen LC, Murphy KP et al. Weakly-and semi-supervised learning of a deep convolutional network for semantic image segmentation. In *Proceedings of the IEEE International Conference on Computer Vision*. pp. 1742–1750.
34. Mittal S, Tatarchenko M and Brox T. Semi-supervised semantic segmentation with high-and low-level consistency. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 2019; .
35. Triguero I, García S and Herrera F. Self-labeled techniques for semi-supervised learning: taxonomy, software and empirical study. *Knowledge and Information systems* 2015; 42(2): 245–284.
36. Hung WC, Tsai YH, Liou YT et al. Adversarial learning for semi-supervised semantic segmentation. *arXiv preprint arXiv:180207934* 2018; .
37. Misra I, Shrivastava A and Hebert M. Watch and learn: Semi-supervised learning for object detectors from video. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. pp. 3593–3602.
38. Settles B. Active learning. *Synthesis lectures on artificial intelligence and machine learning* 2012; 6(1): 1–114.
39. Tian Y, Cheng G, Gelernter J et al. Joint temporal context exploitation and active learning for video segmentation. *Pattern Recognition* 2020; 100: 107158.
40. Beluch WH, Genewein T, Nürnberger A et al. The power of ensembles for active learning in image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. pp. 9368–9377.
41. Sener O and Savarese S. Active learning for convolutional neural networks: A core-set approach. In *International Conference on Learning Representations*. URL https://openreview.net/forum?id=H1aIuk-RW.
42. Morrison D, Milan A and Antonakos E. Uncertainty-aware instance segmentation using dropout sampling. In *CVPR*

workshop on the Robotic Vision Probabilistic Object Detection Challenge.

43. Gal Y, Islam R and Ghahramani Z. Deep bayesian active learning with image data. In *International Conference on Machine Learning*. PMLR, pp. 1183–1192.
44. Yang L, Zhang Y, Chen J et al. Suggestive annotation: A deep active learning framework for biomedical image segmentation. In *International conference on medical image computing and computer-assisted intervention*. Springer, pp. 399–407.
45. Siddiqui Y, Valentin J and Nießner M. Viewal: Active learning with viewpoint entropy for semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. pp. 9433–9443.
46. Goodfellow I, Bengio Y and Courville A. *Deep learning*. MIT press, 2016.
47. He K, Zhang X, Ren S et al. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. pp. 770–778.
48. Ghen G. Bridge inspection robot deployment systems (BIRDS). Technical report, Department of Transportation (Technical Report: 69A3551747126), 2020. URL https://rip.trb.org/view/1596887.
49. Dutta A and Zisserman A. The VIA annotation software for images, audio and video. In *Proceedings of the 27th ACM International Conference on Multimedia*. MM '19, New York, NY, USA: ACM. ISBN 978-1-4503-6889-6/19/10, pp. 2276–2279. DOI:10.1145/3343031.3350535. URL https://doi.org/10.1145/3343031.3350535.
50. Abdulla W. Mask R-CNN for object detection and instance segmentation on Keras and TensorFlow, 2017. URL https://github.com/matterport/Mask_RCNN.