Proceedings of the 2020 Winter Simulation Conference
K.-H. Bae, B. Feng, S. Kim, S. Lazarova-Molnar, Z. Zheng, T. Roeder, and R. Thiesing, eds.

DEEP GENERATIVE ADVERSARIAL NETWORK TO ENHANCE IMAGE QUALITY FOR FAST OBJECT DETECTION IN CONSTRUCTION SITES

Nipun Nath

Zachry Department of Civil Engineering
Texas A&M University
3136 TAMU
College Station, TX 77843, USA

Amir H. Behzadan

Department of Construction Science
Texas A&M University
3137 TAMU
College Station, TX 77843, USA

ABSTRACT

Visual recognition of the content and actions that take place in a construction site is important in many applications such as data-driven simulation, autonomous systems, and intelligent machinery. Construction project, however, are dynamic and complex, and often take place in harsh environments. This may hinder the ability to collect good quality, well-lit, and occlusion-free imagery, which in turn, can lower the performance of computer vision models for fast and reliable object detection. In this paper, we propose and validate a deep convolutional neural network (CNN)-based generative adversarial network (GAN) trained and tested on construction site photos from two in-house datasets to increase image resolution by generating missing pixel information. Results show that using GAN-enhanced images can improve the average precision of pre-trained models for detecting objects such as building, equipment, worker, hard hat, and safety vest by up to 32% while maintaining the overall processing time for real-time object detection.

1 INTRODUCTION

The digital image is one of the most common media to document construction fieldwork. In recent years, the ubiquity of digital cameras, mobile devices (e.g., smartphone and tablet computer) with internet connectivity, and unmanned aerial vehicles (UAVs), also known as drones, equipped with onboard cameras has exponentially increased the volume of visual data collected on a daily basis. By some estimates, more than 1 trillion images were taken in 2018 (The Conversation 2018), which is 50% more than the 657 billion taken only four years prior to that, in 2014 (Kleiner Perkins 2014). In particular to construction, visual data can be used to generate progress reports and requests for information (RFIs), conduct quality inspection, monitor crew productivity, manage resource deployment, perform safety training, and litigate claims.

Timely and reliable harvesting and delivery of information from this big visual data requires significant long-term investments in skilled human resources and computing infrastructure (Business Higher Education Forum 2017). Meanwhile, artificial intelligence (AI) and its latest derivatives such as hybrid intelligence (a.k.a., human-in-the-loop AI) (Kamar and Redmond 2016) hold promise for expanding human knowledge, especially when confronted with large amounts of data. Examples of using AI-assisted tools for analyzing construction visual data, particularly those using vision-based algorithms, include content retrieval from site photos (Nath et al. 2019), identifying construction materials and resources (Dimitrov and Golparvar-Fard 2014), monitoring worker safety (Nath et al. 2020; Kim et al. 2019), and 3D reconstruction of infrastructure (Brilakis et al. 2011). In these and similar applications, object detection (i.e., identifying the location and category of objects in a given image) is one of the most common tasks, and of particular interest to machine-driven autonomous systems. For example, in the fully-automated construction site of the future, unmanned vehicles must identify obstacles and calculate an accident-free path to their destinations. Similarly, construction robots tasked with lifting concrete

blocks, tying rebar, or laying bricks must first identify the correct objects (e.g., concrete block, rebar, brick) to work with. Moreover, an AI model trained with both object detection and human behavior coding can alert workers of unsafe actions or imminent accidents. Lastly, to monitor material inventory, an AI-enabled vision system should be able to recognize the quantity and layout of warehouse objects from video surveillance.

All vision-based algorithms (including object detection) perform remarkably better when the quality of the input image is high. Past studies have shown that poorly-textured objects in the images captured by a stereo camera can lead to the generation of unreliable depth maps and found that high-resolution images provide more key points to accurately calculate the disparities between the stereo images (Geiger et al. 2010). Researchers have achieved a 30% improvement in facial recognition by deblurring low-quality photos (Li et al. 2018). Similarly, in medical imaging, high-resolution imagery is desirable to retrieve vital biological, anatomical, physical, and metabolic information which might be difficult to catch in a low-resolution (noisy or blurry) image (Trinh et al. 2014). Another area where higher resolution images capture more crucial pieces of evidence for future investigations include video surveillance (e.g., for public security, traffic monitoring, military reconnaissance) (Kumar et al. 2016).

High quality visual data could be also very important for creating data-driven models to simulate ongoing jobsite operations. Since high-resolution images contain richer information and insight about the real-world, they can help close the gap between simulation and reality. In the construction domain, for example, high quality images have been used to model digital twin of a construction site to monitor spatiotemporal activities, analyze physical vulnerabilities, and optimize the spatial layout and workflows to increase productivity and minimize potential risks (Ham and Kim, 2020). In another example, high fidelity images were used to create augmented reality (AR) and virtual reality (VR) simulations for advanced construction management (Ahmed, 2018).

Despite the advantages of high-resolution images in many applications, it may not be always practical to obtain such images due to the limitation of hardware, cost of acquiring and operating large-scale image capturing devices, and adversarial surrounding environment (Yue et al. 2016). For example, in a construction site, given the dynamic and complex workflow, depending on camera position and angle, photos may contain several small objects at various distances and poses. Therefore, even if a scene is captured with high-resolution, the number of pixels corresponding to some objects (especially those that are smaller in size, farther from the camera, or partially occluded by other objects) might be too few, which can limit the performance of object detection.

One way to overcome this challenge is to enlarge the entire image or some parts of it (a.k.a. up-sampling) by interpolating the intermediate pixels, for example, using bilinear, bicubic or Lanczos filtering (Shan et al. 2008). However, these methods rely only on the local information stored in the low-resolution image, and as such, may generate exceedingly blurry images that fail to preserve critical features (e.g., textures and edges) for object detection (Shan et al. 2008). On the contrary, example-based methods, that rely on training images, learn a general process to restore missing pixels with rich and fine details based on the spatial contexts in the given low-resolution image (Freeman et al. 2002). To this end, deep learning-based methods have achieved remarkable performance in recent days (Ledig et al. 2017). Particularly, past research has found that generative adversarial network (GAN) is more reliable in producing realistic and natural up-sampled images (Ledig et al. 2017).

In this paper, the authors investigate a GAN-based method to enhance the resolution (and quality) of an input image. By post-processing low-resolution images, this method eliminates the need for additional hardware resources, thus lowering the overall time and cost. The developed model learns what key information is generally hidden in the low-resolution input image and reconstructs an enlarged version of the image by interpolating missing information. Building upon the authors' past work, in this paper, the performance of the model is assessed by testing it on images containing construction-related objects (e.g., building, equipment, worker, hard hat, safety vest).

2 PROBLEM STATEMENT

This paper utilizes a GAN technique for reconstructing higher resolution versions (a.k.a., super-resolved) of low-resolution input photos taken from construction sites. The performance of this technique in detecting objects in low-resolution images and their GAN-generated super-resolved images without additional training is measured using two you-only-look-once (YOLO) (Redmon and Farhadi 2018) models previously developed by the authors. The first model is trained and tested to detect common construction objects, e.g., building, equipment, and worker (Nath and Behzadan 2020), while the second model is designed for monitoring personal protective equipment (PPE) compliance by detecting hard hat, safety vest, and worker (Nath et al. 2020). It will be demonstrated that GAN and YOLO, together, can process an image in high-speed (i.e., <200 milliseconds, or >5 frames per second, FPS) on a GPU-equipped Dell XPS 7590 laptop.

3 GAN MODEL FOR IMPROVING IMAGE QUALITY

A GAN model has two components, a generator G and a discriminator D (Goodfellow et al. 2014). In the problem at hand, given a low quality or low-resolution image I^L, the objective of G is to generate the same image but with higher resolution and enhanced quality, a.k.a. super-resolved image, I^S (Ledig et al. 2017). However, each low-resolution image (I^L) has a ground-truth high quality counterpart image (I^H). The goal of D is to accurately distinguish between the I^S (generated) and I^H (real) images. During training, G and D compete with one another to improve their performance (Goodfellow et al. 2014). Particularly, G tries to generate I^S images similar to I^H images so that D fails to catch the differences. On the other hand, D tries to improve its ability to learn more subtle differences between I^S and I^H images so that G cannot deceive it. From the perspective of game theory (Freund and Schapire 1996), this constitutes a minimax game where G and D are two agents and the game settles when each agent achieves the minimum level of competency that is perceived as maximum by the other agent (Goodfellow et al. 2014). The training process is known as adversarial training (Goodfellow et al. 2014) with the expectation that at the end of the training, a discriminator D is obtained which can differentiate I^S and I^H images with human-level accuracy. Equally important is a generator G which is capable of generating high-quality I^S images that are difficult to distinguish from the I^H images by the high-performing D, and with the same token, by a human.

3.1 The Architecture of the Generator and Discriminator Models

Figure 1 illustrates the structure of the generator (G) and discriminator (D). As shown in the Figure, the generator is a fully-convolutional autoencoder, based on residual blocks (He et al. 2016), which performs a series of convolutions, batch normalizations, additions, and activations, e.g., leaky rectified linear unit (ReLU) and parametric ReLU (PReLU), on the input image (Ledig et al. 2017). While the other layers keep the dimensions of the input image unchanged after performing operations, the up-sampling layer doubles the height and width of the input image. Thus, the two up-sampling layers in G (Figure 1), collectively increase the image dimensions by a factor of 4. Overall, the generator takes a low-resolution image, I^L, and generates 4×4 times higher resolution version of it, called I^S. On the other hand, the discriminator (D) is based on the VGG network containing both convolutional and fully-connected (a.k.a. dense) layers (Simonyan and Zisserman 2014). This network takes a high-resolution image, I^S (generated) or I^H (real), and outputs the probability of the image being real (I^H). In other words, the discriminator performs a binary classification of the input image.

To note, the fully-convolutional functionality of the generator provides flexibility to operate on an arbitrarily-sized input image. Therefore, given an input image I^L of size $w \times h$, the network will generate image I^S with size $4w \times 4h$. However, the dense layers in the discriminator mandate that the size of the input be fixed. Moreover, the discriminator is only used in the training phase. Therefore, during training, I^L of 48×48 resolution is used, which means that the resolution of I^S and I^H would be 192×192 in the

training phase. However, in the testing phase, the restriction on the input size is removed and different input resolutions, e.g., 52×52, 72×72, 96×96, 144×144, and 208×208, for I^L are investigated.

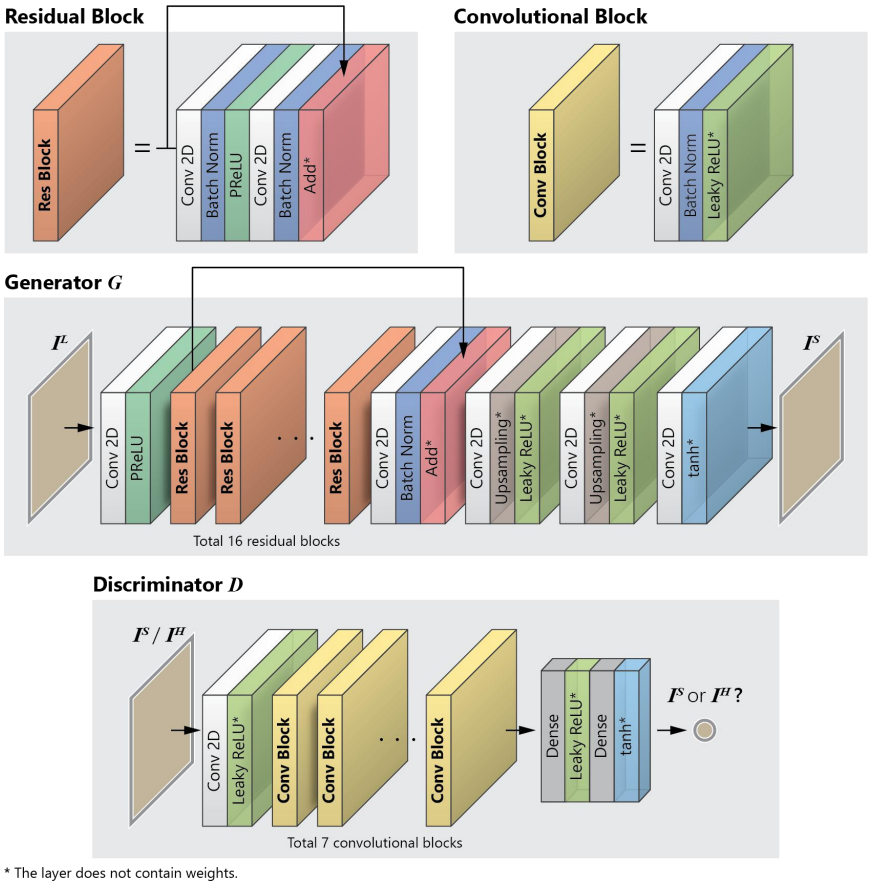

Figure 1: Architecture of the generator and discriminator networks, inspired by Ledig et al. (2017).

3.2 Loss Functions

To train the generator and discriminator networks, three loss functions, namely the binary cross-entropy, content loss, and perceptual loss are used (Ledig et al. 2017). In particular, given an image I^H ($y^{True} = 1$) or I^S ($y^{True} = 0$), if the discriminator predicts the image as I^H with the probability of y^{Pred}, the binary cross-entropy loss (Aggarwal 2018) is defined by Equation (1).

$$L_{\text{binary}} = -\left[y^{True} \log\left(y^{Pred}\right) + \left(1 - y^{True}\right) \log\left(1 - y^{Pred}\right)\right] \tag{1}$$

To calculate content loss, a VGG-19 model, without the fully-connected layers, and pre-trained on the ImageNet dataset, is used (Simonyan and Zisserman 2014). Given the I^S and I^H images, the network extracts the two-dimensional feature maps F^S and F^H, respectively, each of size $w_F \times h_F$. The content loss is then defined as the Euclidean distance between these two feature maps (Ledig et al. 2017), as expressed in Equation (2).

$$L_{\text{content}} = \frac{1}{w_F h_F} \sum_{i=1}^{w_F} \sum_{j=1}^{h_F} \left(F^H_{i,j} - F^S_{i,j}\right)^2 \tag{2}$$

Finally, the perceptual loss is defined as the weighted average of content loss and binary cross-entropy loss (Ledig et al. 2017), as shown in Equation (2).

$$L_{\text{perceptual}} = L_{\text{content}} + 10^{-3} L_{\text{binary}} \tag{3}$$

3.3 Training of the Generator and Discriminator Models

As shown in Figure 2, at each iteration of training, a single batch of training images (I^L) is first fed to the generator (G) and outputs (I^S) are recorded. Next, generated images (I^S) and corresponding ground-truth images (I^H) are assorted and fed to the discriminator (D) to check if it can distinguish between them. This task is analogous to binary classification where the job of the discriminator is to classify any given image into two classes: I^Sor I^H. Based on D's output, the binary cross-entropy loss is calculated using Equation (1), and subsequently, its weights are updated using backpropagation (Aggarwal 2018).

Next, the process is repeated with another batch of I^L images and binary loss is calculated, but without updating D's weights. This time, G's outputs (I^S) and corresponding ground-truths (I^H) are also fed to the VGG-19 model and the content loss is calculated using Equation (2). Based on the VGG-19's content loss and D's binary loss, the perceptual loss is subsequently calculated using Equation (3) and G's weights are updated. This sequential updates of D and G accomplish one iteration of training. The number of iterations in one epoch is equal to the number of training images divided by the number of images in one batch (a.k.a. batch size), rounded to the lowest integer.

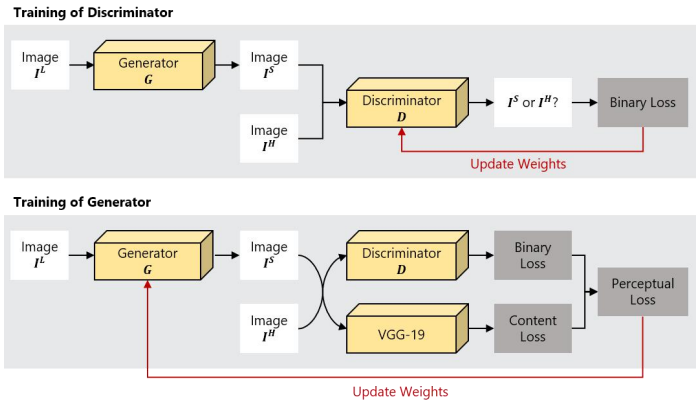

Figure 2: Schematic diagram of one iteration of training GAN models.

4 DATASET DESCRIPTION

This study uses the combination of two previously developed datasets by the authors, Pictor-v2 (Nath and Behzadan 2020) and Pictor-v3 (Nath et al. 2020). As shown in Figure 3, Pictor-v2 contains 1,994 training and 513 testing images, labeled with three object classes – building (B), equipment (E), and worker (W). On the other hand, Pictor-v3 contains 1,184 training and 288 testing images, also labeled with three object classes – hat (H), vest (V), and worker (W). The GAN model will be evaluated based on the performance of two YOLO models, namely YOLO-BEW and YOLO-PPE, which are trained on the training subsets of Pictor-v2 and Pictor-v3 datasets, respectively. To ensure that these YOLO models do not encounter any images on which they are already trained, the union of the testing subsets of Pictor-v2 and Pictor-v3 datasets is used for evaluating the GAN and YOLO models and, therefore, excluded from training the GAN models. This results in a total of 1,906 training and 744 testing images for the GAN models, as shown in Figure 3.

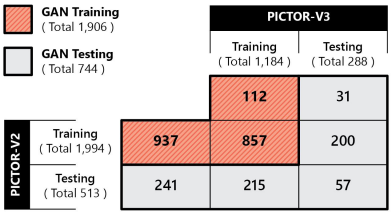

Figure 3: Number of images in the training and testing subsets.

5 METHODOLOGY

5.1 Data Preparation

To prepare the data for training and testing the GAN model, the colors of the images are linearly scaled (a.k.a., normalized) so that all values remain in the range of $[-1, +1]$. Next, images are divided into two subsets of training and testing. Only the training images are randomly augmented, as shown in Figure 4.

Figure 4: Preparation of training data through random augmentation.

First, randomly selected 50% of the original training images are flipped horizontally. Next, color-shifting is performed by multiplying the color of each pixel by a random value drawn from a normal distribution $N(1.0, 0.1)$, and adding another random value drawn from $N(0.0, 0.1)$ distribution. Following this step, $m \times n$ rectangular tiles are generated for each image. For the images with an aspect ratio (i.e., the ratio between image width and its height) between 0.5 to 2.0, $m = 2$ and $n = 2$ are used. For other aspect ratios (i.e., <0.5 or >2.0), 2 rectangular tiles, stacked along the longer dimension of the image, are generated. Next, each tile is randomly scaled by multiplying its box size with a factor drawn from $N(1.0, 0.25)$ distribution. Also, tile centers are further shifted along X and Y directions, each by a length from $N(1.0, 0.167)$ distribution, multiplied by their size along that direction. To note, the use of the

normal distributions is based on Krizhevsky et al. (2017), however, the mean and standard deviation of these distributions are selected empirically by examining the dataset.

During the scaling and translation of the tile boxes, if any box is moved outside the image boundaries, it is trimmed so that the residual part remains inside the image. Next, the portion of the image within each tile box is cropped. Each cropped image is then resized to a 192×192 square-sized image which is treated as the high-resolution (ground-truth) image, or I^H. The I^H is subsequently resized to 48×48 resolution which serves as the low-resolution version, or I^L, corresponding to the I^H. Finally, I^L and I^H images are stacked in batches to allow the GPU to perform operations (e.g., convolutions) on all the images in one batch simultaneously, rather than treating each image individually, thus leading to a significantly lower computational time.

5.2 Training of GAN Models

At the beginning of each epoch, all training images are randomly augmented following the method described in Subsection 5.1. Such random augmentation allows the model to encounter slightly different versions of the training images in each epoch of training. Thus, instead of memorizing the training images, the model tries to learn the latent features of the images. Next, models are trained following the process illustrated in Subsection 3.3. The weights of the D and G are updated through backpropagation using Adam optimizer (Aggarwal 2018) with a starting learning rate of 1×10^{-4}. However, after 150, 250, and 300 epochs, the learning rate is reduced by a factor of 5. Finally, training is terminated after 330 epochs.

5.3 Testing

As mentioned earlier, two object detection models, based on the YOLO algorithm (Redmon and Farhadi 2018) are tested on the low-resolution images (I^L) and super-resolved images (I^S) generated by the trained G model. One model (referred to as YOLO-BEW) is trained on Pictor-v2 dataset to detect common construction objects, namely buildings (B), equipment (E), and workers (W) (Nath and Behzadan 2020). Another model (referred to as YOLO-PPE) is trained on the Pictor-v3 dataset to detect workers (W) and personal protective equipment (PPE), e.g., hat (H) and vest (V) (Nath et al. 2020). Each model takes a 416×416 resolution image and outputs bounding boxes for detected objects.

To investigate the influence of different image resolutions, I^L images are created by letterboxing the original testing images to sizes 52×52, 72×72, 96×96, 144×144, and 208×208 resolutions. Following this step, to test the performance of object detection model on low-resolution images (referred to as Model-LR), I^L images are directly fed to the YOLO models and bounding boxes for detected objects are recorded. On the other hand, to test the performance of object detection model on GAN-improved images (referred to as Model-SR), first, each I^L image is broken down to 2×2 tiles and stacked into one batch of 4 images. Next, the batch is given to the trained G model to generate corresponding 4 super-resolved images, which are then tiled back to create the full image I^S. Finally, this I^S image is supplied to the YOLO models and detected bounding boxes are recorded.

5.4 Performance Evaluation

To evaluate the quality of the generated I^S images, the content loss is calculated using Equation (1) to determine how much useful content is missing in I^S images compared to the I^H (ground-truth) images (Ledig et al. 2017). The lower value of content loss implies higher perceptual similarities between the two images. Additionally, another metric, called blind (or referenceless) image spatial quality evaluator (BRISQUE), is used (Mittal et al. 2012) to evaluates an image as a whole and measure the possible loss of naturalness in it. Similar to content loss, the lower score of BRISQUE indicates better quality of the image. Finally, the performance of YOLO object detection models is measured by calculating average precision (AP) for each class and taking the average of these, a.k.a., mean AP (mAP) (Nath et al. 2020).

6 RESULTS AND DISCUSSION

Example of I^L, I^S and I^H images are shown in Figure 5 and results are discussed in the following Subsections.

Figure 5: Examples of low-resolution (I^L), GAN-generated (I^S), and high-resolution (I^H) images.

6.1 Evaluation of Quality of the Generated Images

Table 1 lists the content loss and BRISQUE score of the I^L and I^S images for different resolutions. The Table shows that the higher the input resolution the lower the content loss. Intuitively, the higher resolution images preserve the contents of the original image. It can be seen that for 52×52, 72×72, and 96×96 images, I^S images have lower content loss than the corresponding I^L images, indicating that some of the contents lost in the I^L images (when resized from I^H images) are successfully retrieved by the GAN model in the I^S images. However, for higher resolutions, i.e., 144×144 and 208×208, I^L images have slightly better contents than the I^S images. One possible reason is that the GAN model is trained on low-resolution (48×48) images and, therefore, performs better on those images. The BRISQUE scores in Table 1 show that I^S images have smaller score (or distortion) and thus, higher naturalness compared to

the I^L images. Moreover, the score does not vary much with the change in input resolution, indicating a consistent level of naturalness in the I^S images.

Table 1: Content loss and BRISQUE score of the low-resolution (I^L) and GAN-improved (I^S) images (lower is better).

Input Size	**52×52**		**72×72**		**96×96**		**144×144**		**208×208**	
Model	I^L	I^S	I^L	I^S	I^L	I^S	I^L	I^S	I^L	I^S
Content Loss	61.3	57.2	51.0	48.5	40.8	39.2	23.7	26.4	12.2	16.0
BRISQUE	85.4	43.8	75.0	45.1	68.5	45.4	65.0	45.7	60.9	46.1

6.2 Performance of Object Detection

The performances of Model-LR and Model-SR for different resolutions of I^L and I^S images are summarized in Table 2 and Table 3, which show that, in general, the SR models outperform the LR models. There is only one case, i.e., the detection of vest (V) in 208×208 input images, where the AP of SR model is almost the same as the LR model (i.e., 83%). However, for all other cases, the AP of the SR model is 2% to 32% better than the corresponding LR model. This finding unequivocally indicates that GAN improves the quality of the image for better object recognition.

Table 2: Performance of Model-LR and Model-SR in detecting building, equipment, and worker in Pictor-v2 dataset.

Input Size	**52×52**		**72×72**		**96×96**		**144×144**		**208×208**	
Model	**LR**	**SR**	**LR**	**SR**	**LR**	**SR**	**LR**	**SR**	**LR**	**SR**
Mean AP (%)	8	22	19	32	30	40	45	51	58	62
Building AP (%)	8	20	18	28	28	32	41	46	54	56
Equipment AP (%)	8	16	15	26	24	38	41	50	60	65
Worker (%)	8	29	24	41	38	48	52	58	62	66
Time (ms)	68	104	63	145	64	169	65	432	65	857

Table 3: Performance of Model-LR and Model-SR in detecting hat, vest, and worker in Pictor-v3 dataset.

Input Size	**52×52**		**72×72**		**96×96**		**144×144**		**208×208**	
Model	**LR**	**SR**	**LR**	**SR**	**LR**	**SR**	**LR**	**SR**	**LR**	**SR**
Mean AP (%)	11	29	25	46	43	56	62	69	71	76
Hat AP(%)	8	19	17	30	31	42	49	59	61	68
Vest AP (%)	15	39	32	64	58	72	77	80	83	83
Worker (%)	11	29	26	44	39	55	61	68	70	76
Time (ms)	82	100	61	145	63	171	69	432	64	856

The mAP of the models for Pictor-v2 and Pictor-v3 datasets is illustrated in Figure 6, which shows that with the increase of input size, the performance of both models improves. Particularly, for lower resolution input images, the SR models perform significantly better than the counterpart LR models. For example, for 52×52 images, compared to the LR model, the SR model is 14% better for the Pictor-v2 (BEW) test dataset and 18% better for the Pictor-v3 (PPE) test dataset. However, for higher resolution images, the difference in the performances of LR and SR models slightly drops. For example, for 208×208 images, compared to the LR model, the SR model is only 4% and 5% better when tested on Pictor-v2 (BEW) and Pictor-v3 (PPE) datasets, respectively. This indicates that when the input resolution

is lower, the GAN model can generate significantly better content-rich images. On the contrary, image resolutions that are sufficiently good, leave less improvement room for the GAN model.

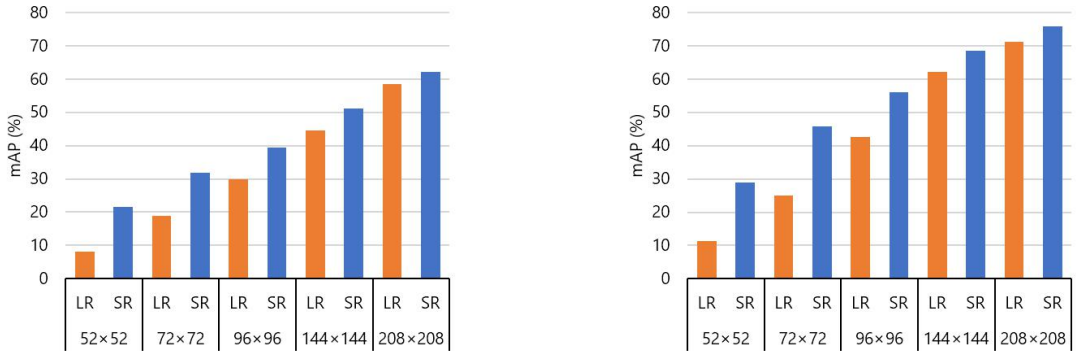

Figure 6: mAP of the LR and SR models for (a) Pictor-v2 (BEW) and (b) Pictor-v3 (PPE) datasets.

Figure 7 shows an example of object detection by YOLO-PPE model for I^L and I^S images with an input size 96×96. The Figure shows that for both images, workers *W1*, *W3*, and *W4*, as well as their PPE components (hat and vest) are detected correctly. However, for worker *W2*, the model missed the hat and incorrectly detected the yellow bucket as a hat in I^L image. Meanwhile, in the I^S image, the model not only did detect *W2*'s hat and vest correctly but also detected the vest with higher confidence (85%) compared to the 49% confidence in the corresponding detection in I^L image. This example highlights some of the primary reasons why YOLO models achieve better mAP on the I^S images than the counterpart I^L images.

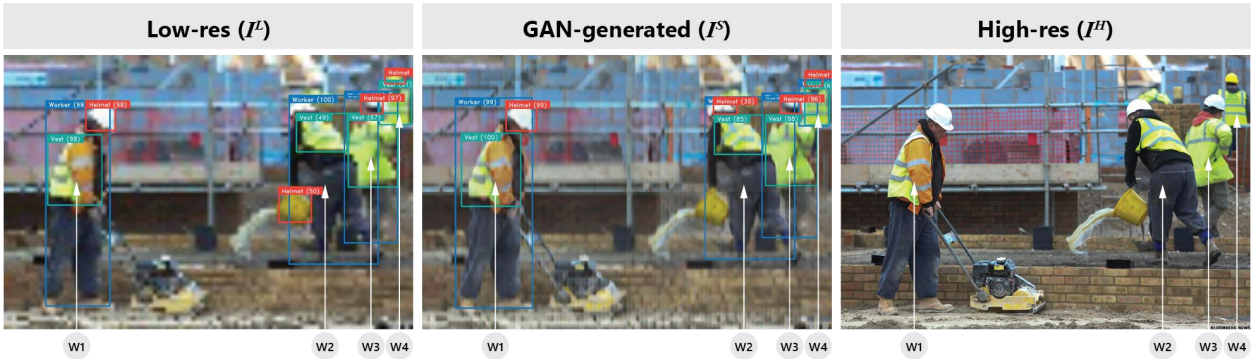

Figure 7: An examples of object detection by YOLO-PPE for low-resolution (I^L) and GAN-generated (I^S) images with an input size of 96×96.

6.3 Processing Times

The average processing times for the LR and SR models, shown in Table 2 and Table 3, indicate that the SR models require more time to process images compared to the LR models. This is expected given that the SR models apply both GAN and YOLO models, while the LR models only apply the YOLO model. However, for 52×52, 72×72, and 96×96 images, the average processing time for one image is less than 200 milliseconds or more than 5 frames per second (FPS), leading to the conclusion that the SR models can be still used for real-time object detection. For the other resolutions, the SR models process images at >1 FPS which is sufficient enough for near real-time object detection.

7 SUMMARY AND CONCLUSION

Although high-resolution images are more reliable (and thus, desirable) for computer vision and simulation applications, in many cases, it may not be possible or practical to obtain such images. For instance, in the construction domain, where jobsites are dynamic and complex, and many projects take place in harsh outdoor environments under varying lighting and atmospheric conditions, taking high quality, well-lit, and occlusion-free imagery of field activities is not a trivial task. Therefore, researchers often rely on post-processing the available low-resolution images before applying computer vision algorithms. In this study, the authors proposed and validated a GAN model to enlarge a low-resolution image by a factor of 4 along its height and width. The proposed model learns from 1,906 training images on how to reconstruct the high-resolution image with rich and fine details. Results show that for 52×52, 72×72, and 96×96 I^L (low-resolution) images, GAN-generated I^S images have less content loss, indicating that the model restored useful contents that were missing in the original input images. Moreover, the low BRISQUE score of the I^S images indicate a higher level of naturalness reconstructed in the generated images.

Two YOLO models, YOLO-BEW trained on Pictor-v2 dataset (building, equipment, worker) and YOLO-PPE trained on Pictor-v3 dataset (hat, vest, worker), were tested on both I^L and I^S images. Results show that in all cases, models tested on the I^S images performed equally or better than those tested on the I^L images. Particularly, for 72×72 input images, YOLO-PPE detected vests in I^S images with 64% AP which is twice the AP of vest detection in I^L images. Overall, these models achieved 4% to 21% better mAP when the input image was improved by GAN, which shows that GAN can improve the quality of low-resolution construction site photos for better object detection. Therefore, the proposed framework can be reliably used to generate high-fidelity input images for vision-based applications including simulation modeling and intelligent machine control. Furthermore, GAN and YOLO models, together, process one image in 100 to 857 milliseconds, making them suitable for real-time object detection.

ACKNOWLEDGMENTS

The authors would like to thank Mr. Yalong Pi, Ph.D. student, for assisting in data preparation. Model training was performed on Texas A&M University's High Performance Research Computing (HPRC) clusters. Any opinions, findings, conclusions, and recommendations expressed in this paper are those of the authors and do not necessarily represent the views of the HPRC or the individual named above.

REFERENCES

Aggarwal, C. C. 2018. *Neural Networks and Deep Learning*. Berlin, Germany: Springer.

Ahmed, S. 2018. "A Review on Using Opportunities of Augmented Reality and Virtual Reality in Construction Project Management", *Organization, Technology and Management in Construction* 10(1): 1839-1852.

Brilakis, I., H. Fathi, and A. Rashidi. 2011. "Progressive 3d Reconstruction of Infrastructure with Videogrammetry", *Automation in Construction* 20(7): 884-95.

The Conversation. 2018. Of the Trillion Photos Taken in 2018, Which Were the Most Memorable? https://theconversation.com/of-the-trillion-photos-taken-in-2018-which-were-the-most-memorable-108815, accessed 23rd April 2020.

Dimitrov, A., and M. Golparvar-Fard. 2014. "Vision-Based Material Recognition for Automated Monitoring of Construction Progress and Generating Building Information Modeling from Unordered Site Image Collections", *Advanced Engineering Informatics* 28(1): 37-49.

Business Higher Education Forum. 2017. Investing in America's Data Science Talent: The Case for Action. https://www.naceweb.org/uploadedfiles/files/2018/publication/free-report/bhef-investing-in-data-science.pdf, accessed 23rd April 2020.

Freeman, W. T., T. R. Jones, and E. C. Pasztor. 2002. "Example-Based Super-Resolution", *IEEE Computer Graphics and Applications* 22(2): 56-65.

Freund, Y., and R. E. Schapire. 1996. "Game Theory, On-Line Prediction and Boosting." In *Proceedings of the Ninth Annual Conference on Computational Learning Theory*, Desenzano del Garda, Italy, 325-32.

Geiger, A., M. Roser, and R. Urtasun. 2010. "Efficient Large-Scale Stereo Matching." In *Proceedings of the Asian Conference on Computer Vision*, Queenstown, New Zealand, 25-38.

Goodfellow, I., J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. 2014. "Generative Adversarial Nets." In *Proceedings of the Advances in Neural Information Processing Systems*, Montréal, Canada, 2672-80.

Ham, Y., and J. Kim. 2020. "Participatory Sensing and Digital Twin City: Updating Virtual City Models for Enhanced Risk-Informed Decision-Making", *Journal of Management in Engineering* 36(3): 04020005.

He, K., X. Zhang, S. Ren, and J. Sun. 2016. "Deep Residual Learning for Image Recognition." In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Las Vegas, NV, 770-78.

Kamar, E., and W. Redmond. 2016. "Hybrid Intelligence and the Future of Work." In *Productivity Decomposed: Getting Big Things Done with Little Microtasks Workshop.* http://research. microsoft. com/en-us/um/people/eckamar/papers/HybridIntelligence.pdf, accessed 23rd April 2020.

Kim, D., M. Liu, S. Lee, and V. R. Kamat. 2019. "Remote Proximity Monitoring between Mobile Construction Resources using Camera-Mounted UAVs", *Automation in Construction* 99: 168-82.

Kleiner Perkins. 2014. 2014 Internet Trends. https://www.kleinerperkins.com/perspectives/2014-internet-trends/, accessed 23rd April 2020.

Krizhevsky, A., I. Sutskever, and G. E. Hinton. 2017. "Imagenet Classification with Deep Convolutional Neural Networks", *Communications of the ACM* 60(6): 84-90.

Kumar, P., A. Singhal, S. Mehta, and A. Mittal. 2016. "Real-Time Moving Object Detection Algorithm on High-Resolution Videos Using GPUs", *Journal of Real-Time Image Processing* 11(1): 93-109.

Ledig, C., L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, and Z. Wang. 2017. "Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network." In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Honolulu, HI, 4681-90.

Li, P., L. Prieto, D. Mery, and P. Flynn. 2018. "Face Recognition in Low Quality Images: A Survey", *arXiv preprint arXiv:1805.11519*.

Mittal, A., A. K. Moorthy, and A. C. Bovik. 2012. "No-Reference Image Quality Assessment in the Spatial Domain", *IEEE Transactions on Image Processing* 21(12): 4695-708.

Nath, N., T. Chaspari, and A. Behzadan. 2019. "Single-and Multi-Label Classification of Construction Objects Using Deep Transfer Learning Methods", *Journal of Information Technology in Construction* 24(28): 511-26.

Nath, N. D., and A. H. Behzadan. 2020. "Deep Convolutional Networks for Construction Object Detection under Different Visual Conditions", *Frontier's in Built Environment* 6(97).

Nath, N. D., A. H. Behzadan, and S. G. Paal. 2020. "Deep Learning for Site Safety: Real-Time Detection of Personal Protective Equipment", *Automation in Construction* 112: 103085.

Redmon, J., and A. Farhadi. 2018. "YOLOv3: An Incremental Improvement", *arXiv preprint arXiv:1804.02767*.

Shan, Q., Z. Li, J. Jia, and C.-K. Tang. 2008. "Fast Image/Video Upsampling", *ACM Transactions on Graphics (TOG)* 27(5): 1-7.

Simonyan, K., and A. Zisserman. 2014. "Very Deep Convolutional Networks for Large-Scale Image Recognition", *arXiv preprint arXiv:1409.1556*.

Trinh, D.-H., M. Luong, F. Dibos, J.-M. Rocchisani, C.-D. Pham, and T. Q. Nguyen. 2014. "Novel Example-Based Method for Super-Resolution and Denoising of Medical Images", *IEEE Transactions on Image Processing* 23(4): 1882-95.

Yue, L., H. Shen, J. Li, Q. Yuan, H. Zhang, and L. Zhang. 2016. "Image Super-Resolution: The Techniques, Applications, and Future", *Signal Processing* 128: 389-408.

AUTHOR BIOGRAPHIES

NIPUN D. NATH is a Ph.D. student in the Zachry Department of Civil Engineering, and is also pursuing an M.S. degree in Computer Science in the Department of Computer Science and Engineering, both at Texas A&M University. He holds an M.S. in Project Management from Missouri State University, and a B.S. in Civil Engineering from Bangladesh University of Engineering and Technology (BUET). His main research interests are machine learning, deep learning, computer vision, and their applications in construction safety and productivity monitoring. His email address is nipundebnath@tamu.edu.

AMIR H. BEHZADAN is an Associate Professor of Construction Science at Texas A&M University. He received his Ph.D. in Civil Engineering in 2008 and his M.Eng. in Construction Engineering and Management in 2005 both from the University of

Michigan, Ann Arbor. He also holds a B.Eng. degree in Civil Engineering from Sharif University of Technology. His research interests include artificial intelligence, machine learning, simulation, and visualization for urban informatics, smart health and ergonomics, disaster resiliency, and safety. He is a member of the American Society of Civil Engineers (ASCE) and serves on the editorial board of the ASCE Journal of Construction Engineering and Management. His email address is abehzadan@tamu.edu.