ITcon

www.itcon.org - Journal of Information Technology in Construction - ISSN 1874-4753

SINGLE- AND MULTI-LABEL CLASSIFICATION OF CONSTRUCTION OBJECTS USING DEEP TRANSFER LEARNING METHODS

SPECIAL ISSUE: **Virtual, Augmented and Mixed: New Realities in Construction**
PUBLISHED: December 2019 at https://www.itcon.org/2019/28
EDITORS: McMeel D. & Gonzalez V. A.
DOI: 10.36680/j.itcon.2019.028

Nipun D. Nath, Ph.D. Student,
Texas A&M University;
nipundebnath@tamu.edu and http://people.tamu.edu/~nipundebnath/

Theodora Chaspari, Assistant Professor,
Texas A&M University;
chaspari@tamu.edu and https://chaspari.engr.tamu.edu/

Amir H. Behzadan, Associate Professor,
Texas A&M University;
abehzadan@tamu.edu and http://people.tamu.edu/~abehzadan/

SUMMARY: *Digital images are extensively used to increase the accuracy and timeliness of progress reports, safety training, requests for information (RFIs), productivity monitoring, and claims and litigation. While these images can be sorted using date and time tags, the task of searching an image dataset for specific visual content is not trivial. In pattern recognition, generating metadata tags describing image contents (objects, scenes) or appearance (colors, context) is referred to as multi-label image annotation. Given the large number and diversity of construction images, it is desirable to generate image tags automatically. Previous work has applied pattern matching to synthetic images or images obtained from constrained settings. In this paper, we present deep learning (particularly, transfer learning) algorithms to annotate construction imagery from unconstrained real-world settings with high fidelity. We propose convolutional neural network (CNN)-based algorithms which take RGB values as input and output the labels of detected objects. Particularly, we have investigated two categories of classification tasks: single-label classification, i.e., a single class (among multiple predefined classes) is assigned to an image, and multi-label classification, i.e., a set of (one or more) classes is assigned to an image. For both cases, the VGG-16 model, pre-trained on the ImageNet dataset, is trained on construction images retrieved with web mining techniques and labeled by human annotators. Testing the trained model on previously unseen photos yields an accuracy of ~90% for single-label classification and ~85% for multi-label classification, indicating the high sensitivity and specificity of the designed methodology in reliably identifying the contents of construction imagery.*

KEYWORDS: *Deep learning, transfer learning, convolutional neural networks, construction photos, web mining, multi-class classification, multi-label classification.*

REFERENCE: *Nipun D. Nath, Theodora Chaspari, Amir H. Behzadan (2019). Single- and multi-label classification of construction objects using deep transfer learning methods. Journal of Information Technology in Construction (ITcon), Special issue: 'Virtual, Augmented and Mixed: New Realities in Construction', Vol. 24, pg. 511-526, DOI: 10.36680/j.itcon.2019.028*

1. INTRODUCTION

Construction site imagery is valuable for creating progress reports, requests for information (RFIs), safety training, productivity monitoring, and claims and litigation. In the advent of digital cameras and more recently, drones, digital images can be readily captured from jobsites and used to increase the accuracy and timeliness of decision-making in construction. However, captured images, despite being abundant, rarely contain rich metadata other than date, time, and (in some cases) location information. Therefore, retrieving desired information or specific visual content from a particularly large image collection may turn into a non-trivial, resource-intensive task that can only be completed manually. A potential remedy to this problem is to create a semantic structure for the image collection, for instance by using metadata tags describing content (e.g., objects, scenes) and appearance (e.g., color, context). However, given the large number and diversity of construction site images, manual tagging is time-consuming and effortful, rendering the automatic generation of metadata an appealing solution.

In pattern recognition, generating metadata tags is referred to as multi-label image annotation. Recent studies have made significant progress in annotating images, i.e., recognizing objects from digital images (Krizhevsky et al., 2012, Simonyan and Zisserman, 2014). However, the majority of such studies aimed at recognizing everyday objects and animals, particularly because of the large number of publicly available datasets. On the contrary, there is only a few publicly available datasets containing construction site images. Previous studies have designed and tested methodologies for recognizing construction equipment, e.g., excavators (Zou and Kim, 2007), and materials (Brilakis and Soibelman, 2008) in digital images. These methodologies follow an extensively careful design of features, while only a few studies utilized deep learning-based automatic feature extraction methods using real-world data (Ding et al., 2018, Kolar et al., 2018, Siddula et al., 2016).

Deep learning methods have achieved significantly promising results in image recognition for large-scale datasets (e.g., with more than 10,000 images). In regards to smaller-scale datasets, however, deep neural networks exhibit several limitations, since the small amount of data cannot be used to adequately learn the large number of weights in the network. For this reason, deep transfer learning approaches have been proposed, which rely on a model pre-trained based on a large-scale dataset from a similar domain to the one of interest, but not necessarily with the same labels to the target task. The weights of the neural network are further retrained based on a new (generally, smaller) dataset (Oquab et al., 2014). Despite potential differences with respect to the image input space and the final class labels, this approach, generally, yields a better result (Oquab et al., 2014), since the network can learn more reliably the basic structure of the data. Therefore, in this paper, the authors present a deep transfer learning-based methodology for annotating construction imagery from unconstrained real-world settings with high fidelity.

Notably, for single-label classification, a convolutional neural network (CNN) is proposed which takes RGB values of an image as input and classifies the image into one of the three categories: *building*, *equipment*, or *worker*. Moreover, for multi-label classification, another CNN-based algorithm is presented that can identify multiple objects of interests (e.g., *building*, *equipment*, *worker*) in the image. The proposed models are trained using construction images that are automatically retrieved using web mining techniques. Finally, the CNN models are tested on unseen images through a validation framework.

2. LITERATURE REVIEW

In computer vision, *image classification* is defined as the problem of assigning a single class (single-label classification) or multiple classes (multi-label classification) to an entire image. With the increase in quantity and quality of photos and videos taken from construction sites, more attention is being drawn to streamlining the process of automatically extracting content from digital imagery through image classification and object detection. For example, Zou and Kim (2007) utilized HSV (hue, saturation, and value) color space of images to identify excavators in construction photos. In particular, they used the threshold of saturation as a feature to distinguish a relatively colorful excavator object from the dark soil or white snow background. Brilakis et al. (2005), and Brilakis and Soibelman (2008) proposed a method to detect shapes in an image and identify corresponding material types (e.g., steel or concrete) within the texture of the detected shape region. Wu et al. (2009) employed Canny edge detection and watershed transformation methods to detect the edges of an object (e.g., columns in an image), and applied object reconstruction to locate and quantify objects (e.g., number of columns). Kim et al. (2016) used scene-parsing and label transfer to match a target image with a number of labeled images, find candidate images that match more closely, and transfer labels from candidate images to the target image.

Recent work has also utilized machine learning (ML) algorithms to automate the process of object recognition in construction site imagery. For example, Chi and Caldas (2011) used naïve Bayes (NB), and neural network (NN) classifiers to detect workers, loaders, and backhoes. Son et al. (2014) used a voting-based ensemble classifier combining several base classifiers such as support vector machine (SVM), NN, NB, decision tree, logistic regression, and k-nearest neighbor (KNN), to identify construction materials (e.g., concrete, steel, and wood) in an image. Dimitrov and Golparvar-Fard (2014), and Han and Golparvar-Fard (2015) used one-vs-all multi-class SVM to classify major construction materials (around 20 types).

The majority of the aforementioned methodologies, however, requires the extraction of handcrafted image features that are particularly relevant to the given classes (Kolar et al., 2018). However, for content-rich imagery such as construction photos that contain a large number of highly diverse objects or cover a large visual field under a variety of environmental conditions (e.g., lighting, landscape, etc.), automatic feature extraction methods such as CNN and histogram of oriented gradients (HOG) are more advantageous because of their ability to self-learn features from a given dataset (Kolar et al., 2018). While HOG poorly performs when high-dimensional features are simultaneously considered for image classification, CNN achieves outstanding results in this task (Kolar et al., 2018) by overcoming the challenge of enormous computational power demanded by traditional NN (LeCun et al., 1998). A good example of CNN can be found in LeCun et al. (1998) which involves recognizing handwritten digits in an image. Other recent studies include but are not limited to classifying 1.2 million images (ImageNet dataset) into 1,000 different classes (various everyday objects and animals such as French fries, printer, umbrella, dog) (Krizhevsky et al., 2012, Simonyan and Zisserman, 2014).

Within the construction domain, there are several studies where CNN has been used for visual analysis of images and videos, mostly for construction safety. For example, Kolar et al. (2018) used CNN to detect safety guardrails in site photos. Siddula et al. (2016) combined the Gaussian mixture model (GMM) with CNN to detect objects of interest in images taken from roof construction sites. Ding et al. (2018) integrated the long short-term memory (LSTM) model with CNN to recognize unsafe behaviors of construction workers (e.g., climbing a ladder) in video frames. More recently, Luo et al. (2018) proposed a method that uses Region-based CNN (R-CNN) to detect 22 classes of construction-related objects and predict construction activities based on the spatial relevance between the detected objects. However, a majority of these object detection (i.e., classifying and localizing objects) algorithms are computationally intensive and require heavy processing power to perform analyses on high volumes of visual data. Moreover, the amount of collected visual data from construction sites is increasing as more contractors rely on reality capture technologies with mobile connectivity such as smartphones, tablet computer, and camera-equipped drones (Ham and Kamari, 2019). For example, a study by Han and Golparvar-Fard (2017) reported that more than 400,000 images were collected during the lifecycle of a 750,000-sf commercial construction project. Therefore, there is a dire need for fast and automated image filtering methods to use data transmission and storage capacities more efficiently. A recent example of existing studies in this direction by Ham and Kamari (2019) uses pixel-by-pixel semantic image segmentation (i.e., assigning a class to each pixel) to train a model to detect construction-related objects. However, manually annotating a large volume of images at pixel-level is a tedious task, requiring substantial amount of time, cost, and human resources and, therefore, might daunt the usability of this method in real practice (Wei et al., 2016). In contrast, image-level annotation is a more practical approach as it requires assigning single or multiple classes to the entire image and, thus, reduces the time and effort to perform manual labeling (Wei et al., 2016). Therefore, considering the advantages of deep learning and image-level annotation, and informed by the need for faster algorithms to process and filter large volumes of visual data for rapid onsite documentation, this research aims at developing a CNN-based methodology to annotate construction site imagery with predefined labels (e.g., building, equipment, and worker). Compared to R-CNN algorithms, the proposed model can be applied in real-time on low-powered mobile devices, i.e., smartphones or drones.

3. DEEP TRANSFER LEARNING

In the following Subsections, the primary building blocks of the developed methodology, i.e., CNN and transfer learning, are briefly described.

3.1 Convolutional neural network (CNN)

Similar to the traditional NN, CNN consists of a series of layers (i.e., input, hidden, and output layers). However, in CNN, the first few hidden layers are convolutional layers where convolution and pooling operations take place (LeCun et al., 2015). Each convolution operation outputs a numerical value by applying a filter (i.e., a matrix of weights) to a sub-region of an image (Kolar et al., 2018). A sample convolution operation involving a 3×3 filter is shown in FIG. 1(a). A pooling operation, on the other hand, is performed to merge semantically similar features into one, thus reducing the size of the image (a.k.a., sub-sampling) (LeCun et al., 2015). FIG. 1(b) illustrates max-pooling, one of the most commonly used pooling operations, where a 2D image is divided into fixed-sized sub-regions (i.e., kernels) and the maximum value in each sub-region is passed to the next layer. The remaining hidden layers are fully-connected layers that are similar to traditional NN. Of note, while working with small training data, to prevent overfitting, some hidden units are often randomly turned off (a.k.a., dropout) (Hinton et al., 2012).

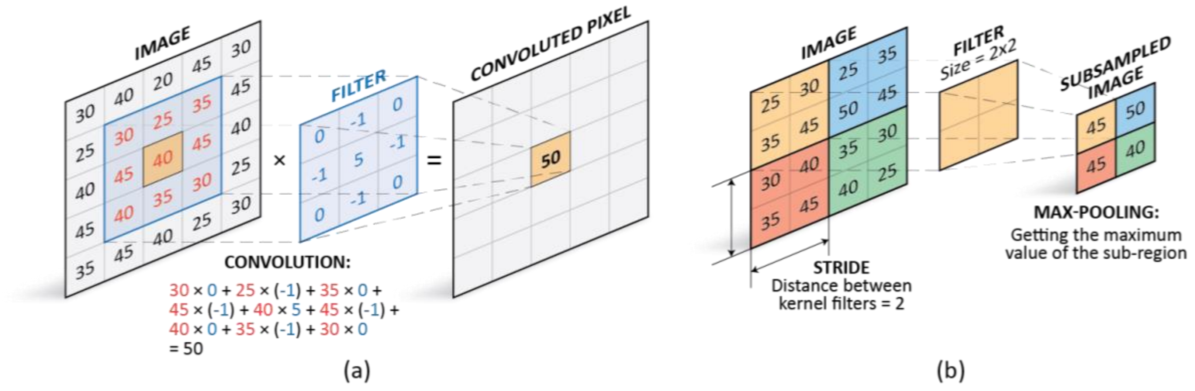
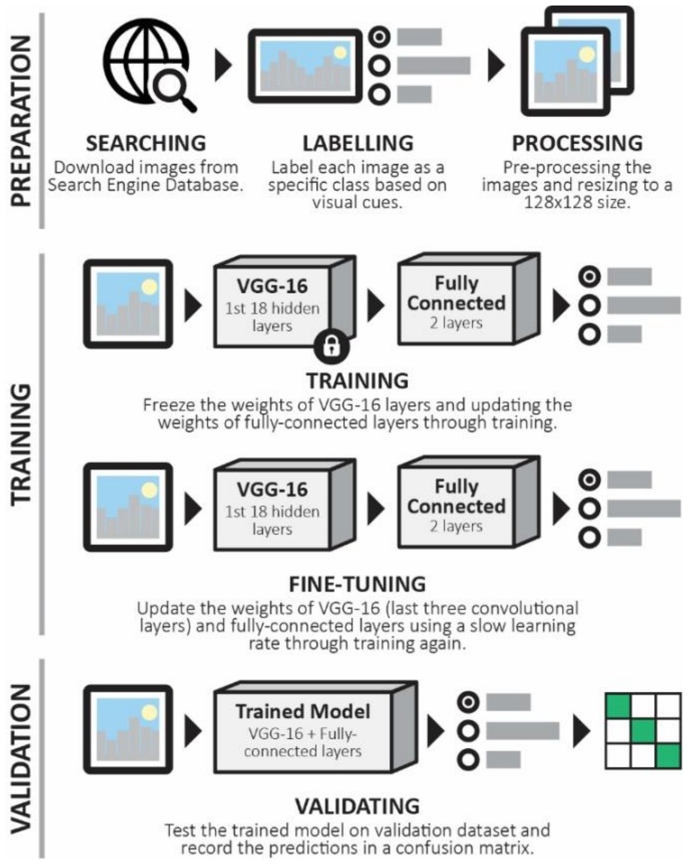


FIG. 1: Example of (a) convolution operation performed with 3×3 filter and (b) max-pooling operation performed with a 2×2 filter.



PREPARATION
SEARCHING
Download images from Search Engine Database.
LABELLING
Label each image as a specific class based on visual cues.
PROCESSING
Pre-processing the images and resizing to a 128x128 size.
TRAINING
VGG-16
1st 18 hidden layers
Fully Connected
2 layers
TRAINING
Freeze the weights of VGG-16 layers and updating the weights of fully-connected layers through training.
FINE-TUNING
Update the weights of VGG-16 (last three convolutional layers) and fully-connected layers using a slow learning rate through training again.
VALIDATION
Trained Model
VGG-16 + Fully-connected layers
VALIDATING
Test the trained model on validation dataset and record the predictions in a confusion matrix.

FIG. 2: Overall framework of the methodology.

3.2 Transfer learning

For a particular dataset, a CNN model can be trained from scratch. However, to achieve optimal results, a large amount of training data coupled with the proper selection of optimal hyper-parameters (e.g., number of layers, number of nodes in each layer, filter size, number of epochs, learning rate, and dropout) is required which might take substantial amount of time for training (Kolar et al., 2018). One way to overcome this challenge is to perform transfer learning, i.e., using a CNN model (e.g., GoogleNet, AlexNet, VGG-16) that is pre-trained with a different but related dataset (a.k.a. source dataset), and partly re-trained with the desired dataset (a.k.a. target dataset). Particularly, transfer learning allows the model to remember high- and mid-level features (e.g., edge, shape, color) learned from the source dataset and apply these features (with minor adjustment) to effectively distinguish the classes in the target dataset (Oquab et al., 2014). Building upon previous studies that have found significantly better and consistent performance using transfer learning (Oquab et al., 2014, Shin et al., 2016), in this research, the authors have used the VGG-16 model, pre-trained on the ImageNet dataset (Simonyan and Zisserman, 2014). The VGG-16 architecture is selected, particularly, for its wide adaptation in various domain, consistent performance comparable to the state-of-the-art techniques (Simonyan and Zisserman, 2014), and manageable size (i.e., only 16 layers of convolution) that allows to port the model on embedded system (e.g., smartphone, drone, autonomous vehicles, portable smart devices) with limited computational power (Alippi et al., 2018).

4. METHODOLOGY

The overall framework of the designed methodology is shown in FIG. 2 and discussed at length in this Section.

4.1 Data preparation

The following Subsections describe the procedure to collect and prepare the image dataset, split the dataset into training and testing subsets, and finally perform data augmentation to generate a larger dataset from a relatively small number of images.

4.1.1 Single-label dataset (Pictor v.1.0)

To obtain sufficient training data for the classifier model, a substantial number of images that contain specific visual contents need to be acquired. One of the most effective tools to achieve this goal is publicly available image search engines which contain a large number of images corresponding to one or more keywords (Fergus et al., 2005). The Google image search database is of particular interest to this research as it contains a relatively large number of images (Deng et al., 2009), and can provide more relevant images with higher ranks (Fergus et al., 2005). In this work, the following keywords are used to search for images in Google: “building under construction”, “construction equipment”, “truck”, “dozer”, “excavator”, “crane”, and “construction worker”. After images are retrieved through web-mining, a web-based labeling toolbox, namely LabelBox (Labelbox, 2019), is used to label all images as containing one of the three possible classes of *building* (building under construction), *equipment* (various construction equipment such as excavator, truck, bulldozer, loader, dozer, and crane), and *worker* (construction worker). These images that are retrieved through web-mining along with the corresponding single-label annotation constitute an in-house dataset called ***Pictor v.1.0*** that is used in this research.

It must be noted that some of the retrieved images could be visually unrelated (Fergus et al., 2005), or manipulated (e.g., the background of construction equipment is removed). In order to obtain a clean image dataset, such irrelevant and manipulated images are labeled as “irrelevant” during the labeling process, and disregarded when preparing the dataset, and training and testing the model. From a total of 2,686 initially retrieved images in ***Pictor v.1.0*** dataset, 2,037 most relevant images are eventually chosen. FIG. 3 shows the number of images and sample images per class label in ***Pictor v.1.0*** dataset.

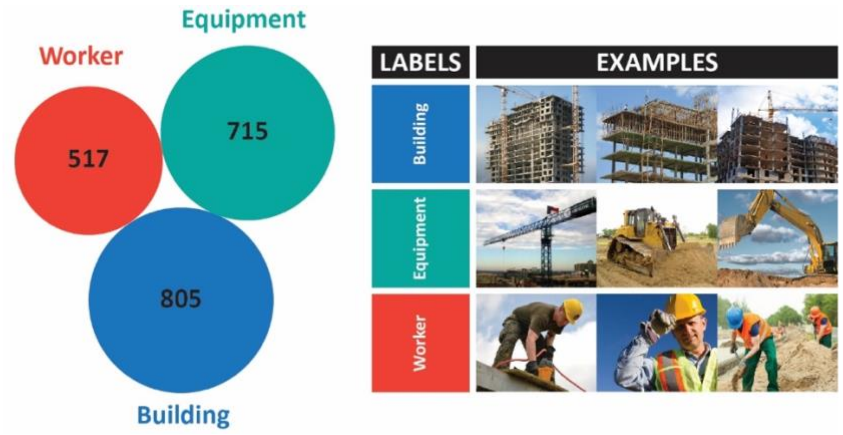


FIG. 3: Number of images and sample images per class label in Pictor v.1.0 dataset.

4.1.2 Multi-label dataset (Pictor v.1.1)

For multi-label classification, ***Pictor v.1.1*** dataset is created by revising the annotations of images in ***Pictor v.1.0*** dataset using Labelbox (Labelbox, 2019). An important distinction between these two datasets is that all images in ***Pictor v.1.1*** dataset contain multi-class labels including *building*, *equipment*, and *worker*. The weighted Venn diagrams in FIG. 4 exhibits the number of images of each class in ***Pictor v.1.1*** dataset.

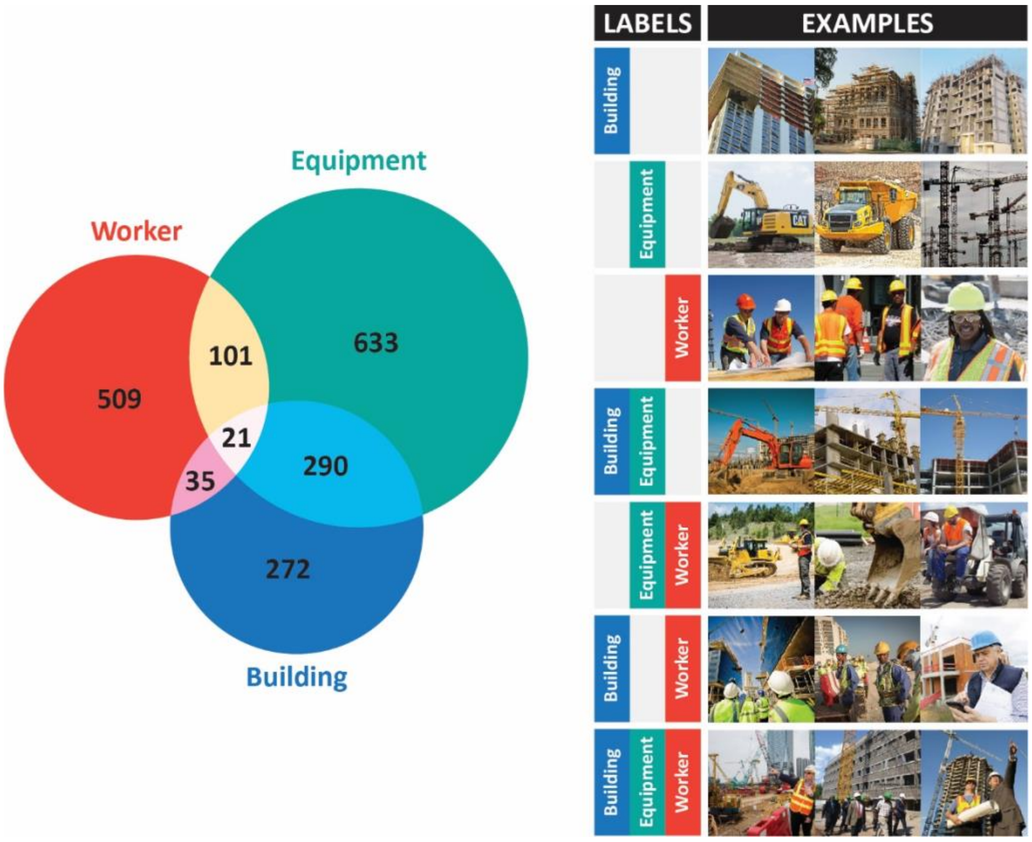


FIG. 4: Number of images and sample images per class label in Pictor v.1.1 dataset.

4.1.3 Data pre-processing and splitting

Since the VGG-16 model only takes square-sized input images, for single-label classification, any rectangular image is cropped into a group of square images that cover the entire visual field of the original image while being equidistantly distributed along the longer dimension of the original image. An example is shown in FIG. 5 where a portrait rectangular image is cropped into three square images. The number of cropped images is determined based on the smallest integer number greater than or equal to (i.e., ceiling of) the ratio between the longer and shorter dimensions of the original image. Next, all cropped images are resized to 128×128 images using the bi-cubic interpolation method (Zhang et al., 2011).

Images in ***Pictor v.1.0*** dataset are cropped into a total of 4,144 square-sized images following the technique previously explained. From these images, 3,392 images (~80%) are randomly selected for training and 752 images (~20%) for testing. The distribution of the number of samples per class label is shown in TABLE 1. As shown in this Table, 1,575 images contain the *building* label, 1,426 images contain the *equipment* label, and 1,143 images contain the *worker* label.

However, for multi-label classification, images in ***Pictor v.1.1*** dataset are not cropped since cropping an image may exclude all the objects of a particular class that the image is labeled with. Rather, the images are resized to 128×128 images using the bi-cubic interpolation method (Zhang et al., 2011). Next, similar to single-label classification, from the entire dataset, randomly selected ~80% samples are used for training while the remaining samples are used for testing. TABLE 2 shows the number of images used for training and testing the multi-label classifier model. As shown in this Table, 479 training images contain building, 841 training images contain equipment, and 539 training images contain worker. In total, 1,589 images are used for training and 398 images are used for testing. It must be noted that since one image may contain multiple classes, the total number of images is not necessarily the sum of the number of images for each class.

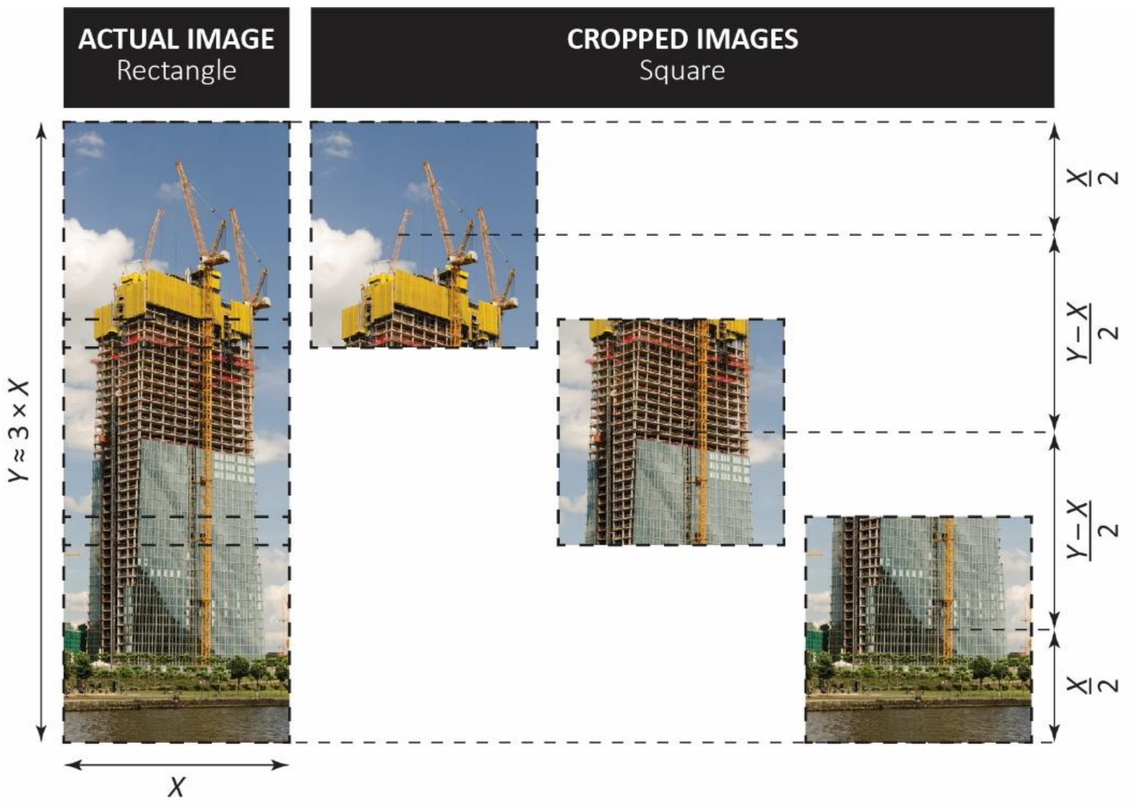


FIG. 5: Example of cropping a rectangular image into a group of square-sized images.

TABLE 1. Number of images in Pictor v.1.0 dataset used for training and testing a single-label classifier model.

	Number of images		
Class	**Train**	**Test**	**Total**
Building	1,284	291	1,575
Equipment	1,160	266	1,426
Worker	948	195	1,143
Total	3,392	752	4,144

TABLE 2. Number of images in Pictor v.1.1 dataset used for training and testing multi-label classifier model.

	Number of images		
Class	**Train**	**Test**	**Total**
Building	479	139	618
Equipment	841	204	1,045
Worker	539	127	666

4.1.4 Data augmentation

Data augmentation is an effective technique to prevent classifier model from overfitting by providing randomly distorted training images to the model and thus, allowing the model to learn general features (Perez and Wang, 2017). In this study, during each epoch of training, training images are distorted by randomly scaling the image by ±20% and horizontally flipping the image randomly 50% of the time. Example of an actual image and randomly generated augmented images are shown in FIG. 6. As shown in this Figure, data augmentation generates more training images with different orientation (e.g., bucket of the excavator facing left and right) and zoom-levels (e.g., the bucket appearing closer in some images, and farther in other images). It allows the model to learn to recognize the objects regardless of their orientation and distance with respect to the camera.

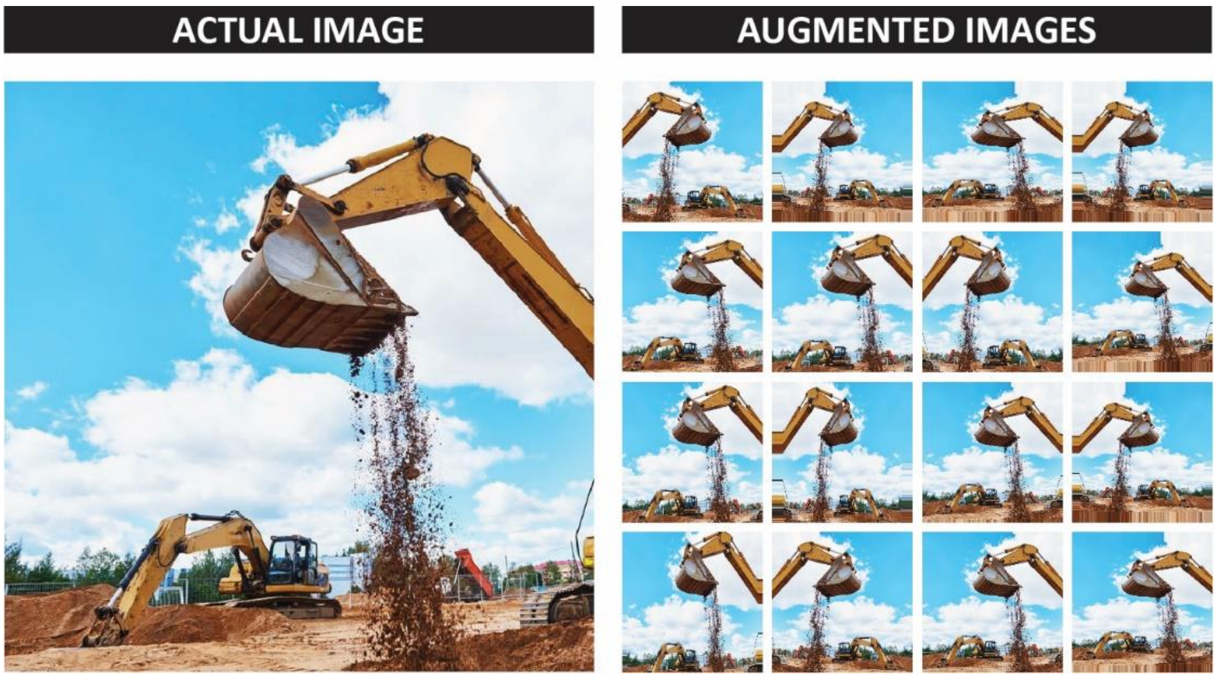


FIG. 6: Example of data augmentation using random scaling and horizontal flipping.

4.2 Model training

The technical details of the CNN architecture, activation and loss functions, and pre-training and fine-tuning the model using transfer learning are described in the following Subsections.

4.2.1 Architecture of the CNN

The designed CNN, for both single-label and multi-label classification, consists of one input layer (i.e., 128×128 RGB images), 18 VGG-16 layers, 2 fully-connected layers, and one output layer (e.g., labels or tags) as shown in FIG. 7. The VGG-16 layers are comprised of a series of convolutional and max-pooling layers with a total number of 14,714,688 pre-trained weights. In the convolutional layers, convolution is performed using a 3×3 filter with a stride of 1 pixel that preserves the size of the image. However, in the pooling layers, max-pooling is performed using a 2×2 filter with a stride of 2 pixels that reduces the size of the image by half in each direction. The outputs of the last VGG-16 layer are connected to a flattened layer consisting of 8,192 nodes, which is fully-connected to the next layer of 256 nodes (the number of nodes in the layer is selected based on empirical observations). In this layer, a dropout operation is performed with 50% probability, i.e., during each iteration of the training session, 50% of the nodes are randomly excluded from weight updating. Together, the two fully-connected layers contain 2,097,408 (i.e., 8192×256) weights. The last hidden-layer is connected to the output layer which yields a vector represented as "one-hot encoding" (Marinai et al., 2005). In this encoding, each element of the vector represents one class and can have a value of either 1 (i.e., the input image belongs to that class) or 0 (i.e., the input image does not belong to that class).

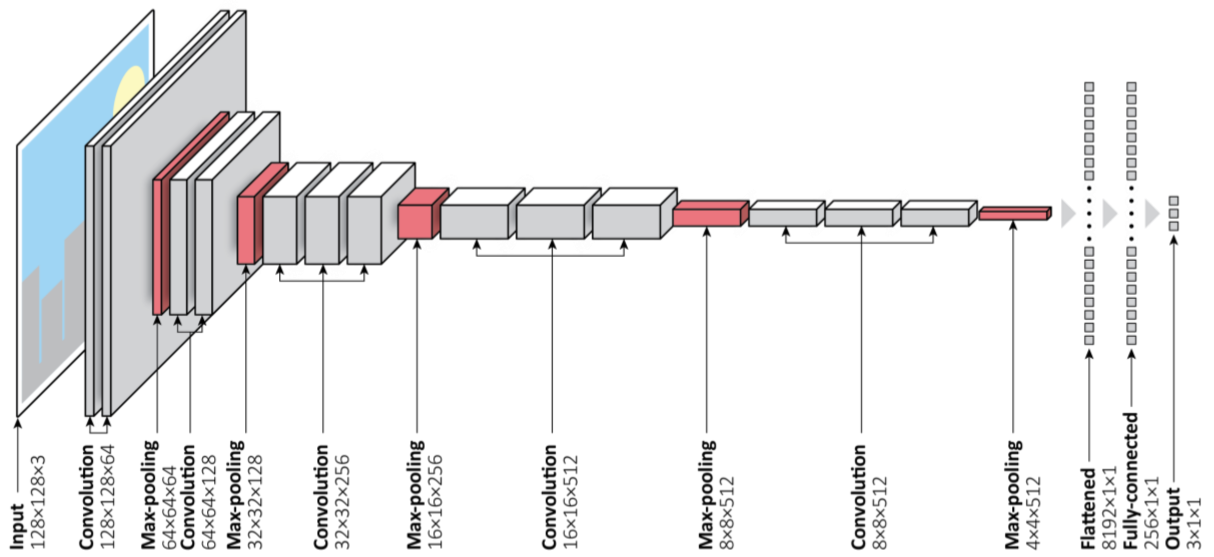


FIG. 7: Architecture of the VGG-16 CNN model.

4.2.2 Activation functions

The rectified linear unit (ReLU) non-linear activation function is applied to the output of each hidden convolutional or fully-connected layer to accelerate convergence (Krizhevsky et al., 2012). While the activation functions are the same at each hidden layer of single-label and multi-label classifier model, they are different at the output layer. For the single-label classifier model, softmax activation function (Murphy, 2012) (Equation 1) is used at the output layer, whereas for the multi-label classifier model, sigmoid activation function (Friedman et al., 2001) (Equation 2) is used.

$$\sigma_{\text{softmax}}(\boldsymbol{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \qquad \text{(Equation 1)}$$

$$\sigma_{\text{sigmoid}}(z_i) = \frac{e^{z_i}}{e^{z_i}+1} \qquad \text{(Equation 2)}$$

Here, z_i is the output value of ith node in the output layer and $\boldsymbol{z}$ is the vector output of the output later, i.e., $\boldsymbol{z} = \{z_1, \dots z_K\}$.

4.2.3 Loss functions

For single-label classification, multi-class cross-entropy (Friedman et al., 2001) is used as loss function. The loss function is defined by Equation 3. On the other hand, for multi-label classification, the loss function is defined as the sum of binary cross-entropy (Buja et al., 2005) over all classes as shown in Equation 4.

$$L_{\text{single\_label}}(\boldsymbol{y},\boldsymbol{p}) = -\sum_{i=1}^{N}\sum_{c=1}^{N_c} y_{i,c}\log(p_{i,c}) \qquad \text{(Equation 3)}$$

$$L_{\text{multi\_label}}(\boldsymbol{y},\boldsymbol{p}) = -\sum_{i=1}^{N}\sum_{c=1}^{N_c}\left[y_{i,c}\log(p_{i,c}) - (1-y_{i,c})\log(1-p_{i,c})\right] \qquad \text{(Equation 4)}$$

Here, N is the total number of samples, N_c is the total number of classes, $y_{i,c}$ and $p_{i,c}$ are the ground-truth label and predicted label, respectively, for the ith sample and cth class, and $\boldsymbol{y}$ and $\boldsymbol{p}$ are matrices containing all the ground-truth and predicted labels, respectively, i.e., $\boldsymbol{y} = [y_{i,c}]$ and $\boldsymbol{p} = [p_{i,c}]$ for $i = 1,2, \dots N$, and $c = 1,2, \dots N_c$. To note, the ground truth labels $(y_{i,c})$ are presented as binary number where one (1) indicates that the sample belongs to the corresponding class, while zero (0) means it does not belong to that class.

4.2.4 Pre-training, re-training, and fine-tuning

For both single-label and multi-label classifications, the VGG-16 is pre-trained on a publicly available dataset (i.e., source data) namely ImageNet dataset (Simonyan and Zisserman, 2014). Next, the training phase is performed in two steps. First, all weights of the VGG-16 layers are frozen from updating, and only the weights of the fully-connected layers are updated using the training dataset. This step allows the CNN model to learn to classify the new set of classes without forgetting the filters learned from the pre-trained dataset. In this step, weights are optimized using the RMSprop optimization algorithm (Tieleman and Hinton, 2012). Next, the training dataset is fed to the model again and weight values of the last three convolutional layers and two fully-connected layers are updated using the stochastic gradient descent (SGD) algorithm (Bottou, 2010) with a slow-learning rate (hyper-parameters, e.g., learning rate = 10^{-4}, and momentum = 0.9, are empirically selected). This step is referred to as "fine-tuning" and allows the previously frozen layers to adapt to the new dataset (i.e., target data) without drastically changing their weights.

4.3 Model testing

To test the performance of the model, unseen testing data are fed to the trained model. The performance of the classifier model in single-label and multi-label classification tasks is evaluated using well-established measures of accuracy, precision, and recall, as shown in Equation 5 through 7.

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \qquad \text{(Equation 5)}$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \qquad \text{(Equation 6)}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \qquad \text{(Equation 7)}$$

Here, TP, TN, FP, and FN refer to true positive (correctly classified to the class), true negative (correctly classified to other class), false positive (incorrectly classified to the class), and false negative (incorrectly classified to other class), respectively. Examples of TP, TN, FP, and FN for the *building* class in single- and multi-label classification are shown in TABLE 3.

TABLE 3. Examples of TP, TN, FP, and FN for the "building" class.

Type	Single-label classification	Multi-label classification
TP	The image is correctly labeled as "building".	Actual image contains building (may contain other classes as well). The model correctly labels it as "building".
TN	Actual image is not labeled as "building. The model does not label it as "building".	Actual image does not contain building (but may contain other classes). The model does not label it as "building".
FP	Actual image is labeled as either "equipment" or "worker". The model incorrectly labels it as "building".	Actual image does not contain building (may contain other classes). The model incorrectly labels it as "building".
FN	Actual image is labeled as "building". The model incorrectly labels it as either "equipment" or "worker".	Actual image contains building (may contain other classes as well). The model does not label it as "building".

5. RESULTS AND DISCUSSION

The designed CNN is applied to ***Pictor v.1.0*** (single-label) and ***Pictor v.1.1*** (multi-label) datasets and the results are demonstrated in the following Subsections.

5.1 Single-label classification results

The CNN model (VGG-16) takes an RGB image as input, generates intermediate features through a series of convolution and max-pooling operations, passes the features to the fully-connected layer, and outputs the probabilities of the image belonging to each class. The intermediate features for a randomly selected image labeled as *building* are shown in FIG. 8. The figure shows that the model finds background sky and edges of the building useful features to detect the building with high probability.

The performance of the single-label classifier model on ***Pictor v1.0*** dataset is summarized in Table 4. Also, classification rates are demonstrated in the confusion matrix of FIG. 9. Table 4 shows that all classes are predicted with ~90% accuracy. Also, the average accuracy, precision, and recall (both weighted and unweighted) are all

>90%. However, the precision of recognizing buildings (i.e., 89.1%) is slightly lower than the other two labels, an indication that it is less likely that an image recognized as building by the model actually contains building(s). Similarly, the recall of recognizing a worker (i.e., 88.7%) is relatively lower than the other two classes, i.e., the model has relatively higher tendency to misclassify an image containing worker as one containing building or equipment. To establish a baseline for the results, a parallel investigation is conducted in which a CNN model is built following a similar architecture to CifarNet (Shin et al., 2016) and trained from scratch using an identical dataset as described earlier in this paper. This model (referred to as "baseline model" in Table 4) yields an accuracy of ~83%, which is ~8% lower than what was ultimately achieved using the pre-trained model (i.e., transfer learning).

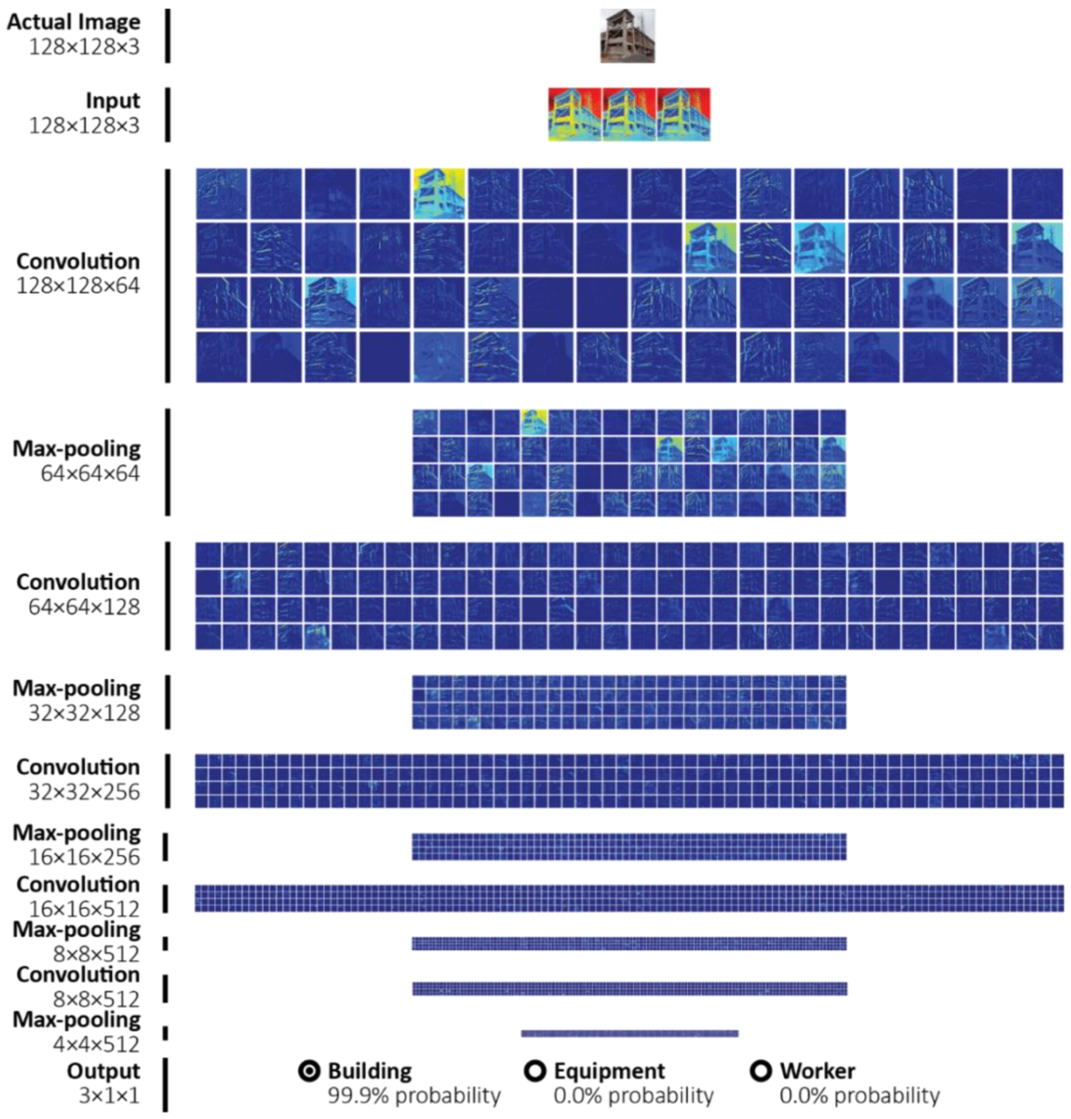


FIG. 8: Visualization of intermediate features provided by the single-label classifier model for an example of a building image.

TABLE 4. Performance metrics of the trained CNN model for single-label classification.

	Designed Model [a]			Baseline Model [b]		
Class	Accuracy	Precision	Recall	Accuracy	Precision	Recall
Building	95.2%	89.1%	95.2%	95.5%	78.3%	95.5%
Equipment	89.5%	92.6%	89.5%	77.4%	87.7%	77.4%
Worker	88.7%	94.0%	88.7%	73.8%	88.9%	73.8%
Unweighted Average	91.1%	91.9%	91.1%	82.3%	85.0%	82.3%
Weighted Average	91.2%	91.3%	91.2%	83.2%	84.1%	83.2%

[a] *VGG-16 fine-tuned on target data, with two fully-connected layers.*
[b] *CifarNet trained from scratch on target data.*

		Prediction		
		Building	Equipment	Worker
Actual	Building	277	9	5
	Equipment	22	238	6
	Worker	12	10	173

FIG. 9: Confusion matrix of the labels predicted by the single-label classifier model.

The underlying reasons behind the misclassifications can be better understood from FIG. 10. In this Figure, the confusion matrix is shown with a few randomly selected sample images. First, it can be seen that most of the misclassified images contain multiple visual cues. Particularly, some that are detected as *building* also contain equipment or worker (or both) in the foreground, obscuring the building in the background. However, the model detects the object in the background that occupies major proportion of the field of vision, rather than the equipment/worker in the foreground, not complying with the labelers' subjective judgment who had labeled those images as *equipment* or *worker*. This justifies the reason behind lower precision in detecting buildings in the image dataset. On the other hand, construction workers are omnipresent and constantly in motion in construction sites especially in the vicinity of buildings or equipment (or both). Therefore, some of the images labeled as *worker* may also contain building and/or equipment. Moreover, the visual footprint of a worker (the portion of the image occupied by a worker) is relatively much smaller than buildings or equipment (e.g., truck, dozer). Thus, the model, having a higher tendency to detect objects with larger visual footprints, may inadvertently mislabel such images as *building* or *equipment*, reducing the recall of detecting workers.

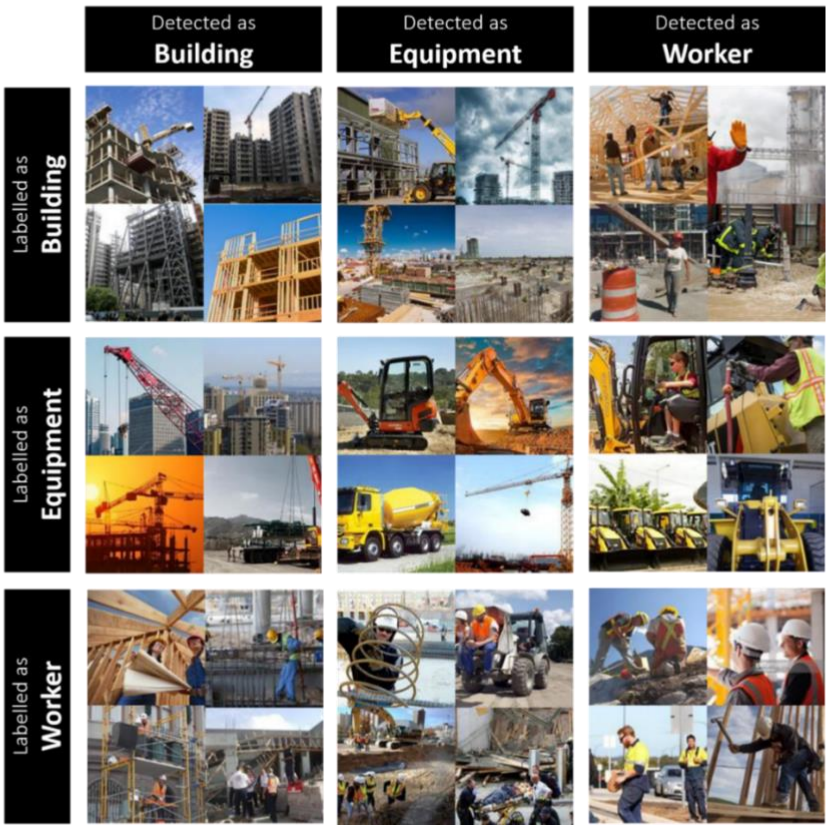


FIG. 10: Visualization of the confused labels in single-label classification.

Despite these issues, the performance of the designed CNN model is still very comparable with the state-of-the-art methodologies. For example, Kolar et al. (2018)'s CNN model detects safety guardrails in images with 86% accuracy (precision = 94.9%, recall = 76.1%). Siddula et al. (2016)'s combined GMM+CNN model detects objects with 96.67% accuracy from roof construction site imagery. Ding et al. (2018)'s CNN+LSTM hybrid model recognizes unsafe climbing of workers with 97% and 92% accuracy while classifying the behavior with two and four labels, respectively. Son et al. (2014)'s SVM model identifies construction materials with 91.68% accuracy. Han and Golparvar-Fard (2015)'s multi-class SVM can classify different construction materials with 92.4% accuracy on average.

5.2 Multi-label classification results

The intermediate features for a randomly selected image labeled as *equipment* and *worker* are shown in FIG. 11. This Figure shows that the model finds the color of the equipment, worker's helmet and vest, and the ground as distinctive features to classify the image as *equipment* and *worker* with high probability.

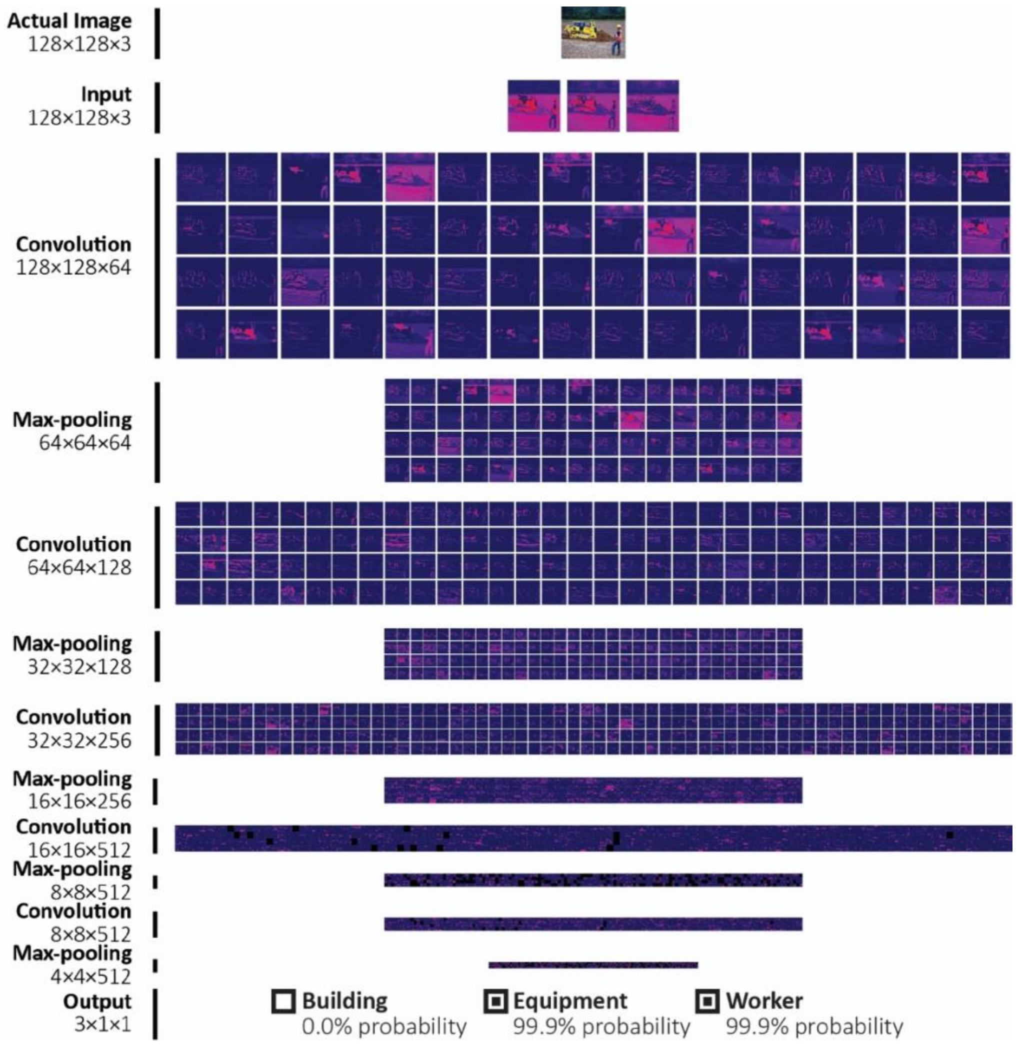


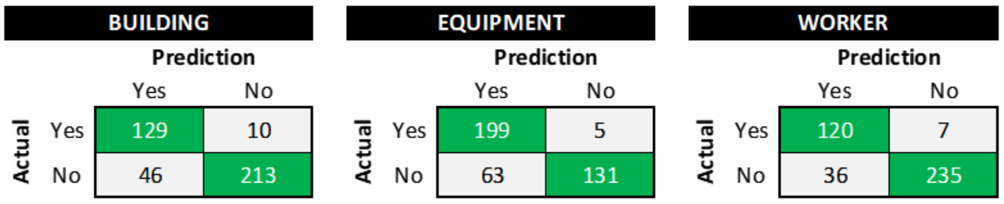
FIG. 11: Visualization of intermediate features provided by the multi-label classifier model for an example of an image containing equipment and worker.

The performance of the multi-label classifier model on ***Pictor v1.1*** dataset is summarized in Table 5. This Table shows that all object classes are detected with >80% accuracy. Also, the average precision and recall (both weighted and unweighted) is >75% and >90%, respectively. A lower precision indicates that sometimes the model predicts the class that is not present in the image. On the other hand, a higher recall indicates that if a class is present in the image, there is a high probability that the model will predict that class. The relatively higher recall of *equipment* class (97.5%) could be attributed to the high number of training samples for that class (TABLE 2). Whereas the relatively higher precision (76.9%) and accuracy (89.2%) of *worker* class could be attributed to the pre-training on the source dataset (ImageNet) which contains visually-related classes (e.g., hat, helmet, jacket, and vest) and thus, allowing the model to effectively transfer the learning of intermediate features to the target dataset (***Pictor v1.1***).

TABLE 5: Performance metrics of the trained CNN model for multi-label classification.

	Designed Model		
Class	Accuracy	Precision	Recall
Building	85.9%	73.7%	92.8%
Equipment	82.9%	76.0%	97.5%
Worker	89.2%	76.9%	94.5%
Unweighted Average	86.0%	75.5%	94.9%
Weighted Average	85.5%	75.6%	95.3%

The classification rates of the designed CNN model are shown in the confusion matrices of FIG. 12, and examples are visualized in FIG. 13. The underlying reasons behind the misclassifications can be discovered from FIG. 13. For example, the Figure shows that some of the images labeled as *building* are not detected as *building*, i.e., false negative (FN), probably because either the buildings are too small in those images or they are occluded by equipment and/or workers. In both cases, it is challenging to extract prominent visual features of the building, leading to a low confidence in classifying these images as *building*. The sample images illustrated in FIG. 13 also show that similar reasons (small visual footprint and occlusion) might be responsible for FN detections of *equipment* and *worker* classes as well. However, in all cases, the rate of FN detections are <8%.



BUILDING

Actual \ Prediction	Yes	No
Yes	129	10
No	46	213

EQUIPMENT

Actual \ Prediction	Yes	No
Yes	199	5
No	63	131

WORKER

Actual \ Prediction	Yes	No
Yes	120	7
No	36	235

FIG. 12: Confusion matrices of the labels predicted by the multi-label classifier model.

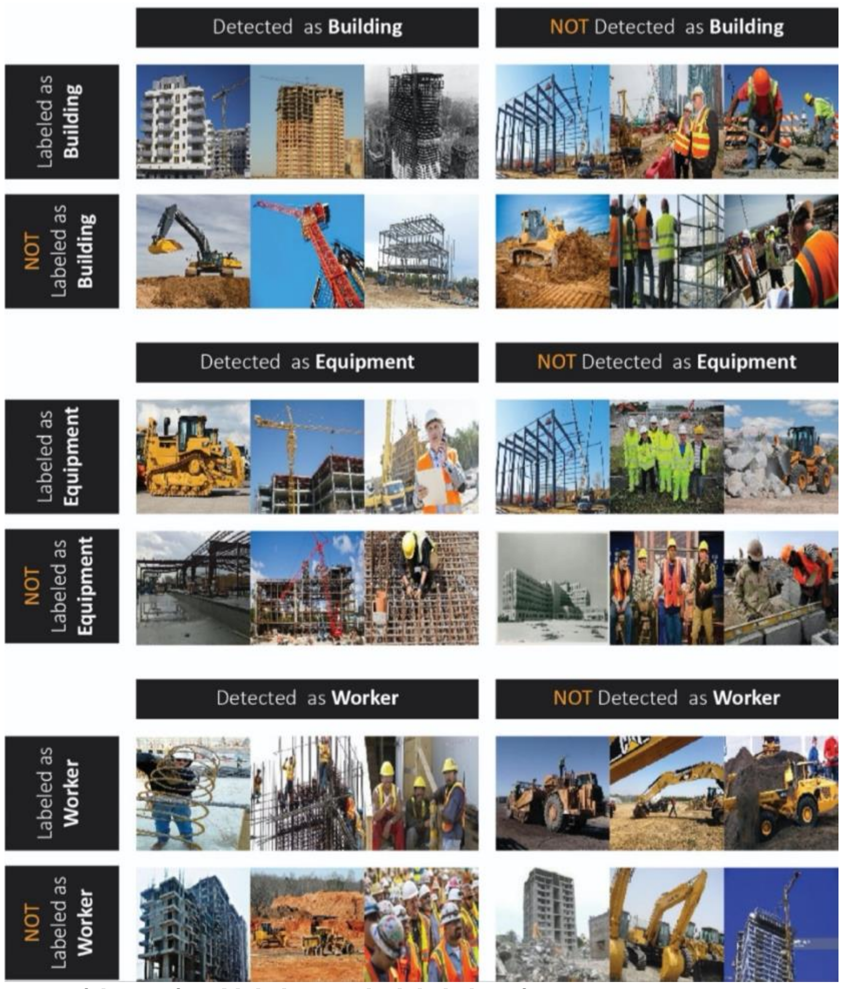


FIG. 13: Visualization of the confused labels in multi-label classification.

The rate of false positive (FP) for *building*, *equipment*, and *worker* classes are 18%, 32%, and 13%, respectively. The high rate of FP for *equipment* could be attributed to the fact that ***Pictor v.1.1*** dataset contains a relatively higher number of images of this class which might bias the model toward falsely predicting the *equipment* class even if it is not present in the image. Nonetheless, a general reason for FP detections of all classes is that some of the images are incorrectly not labeled (by human annotator) as a particular class, even if that class is actually present in those images. Therefore, the model is penalized (both in training and testing) although it performs better than the human annotator in those cases.

6. CONCLUSIONS AND FUTURE WORK

In this paper, deep learning-based methods were introduced that can automatically recognize scene contents (e.g., building under construction) and objects (e.g., construction equipment, construction worker) in RGB images. Two image datasets, namely ***Pictor v.1.0*** (single-label annotation) and ***Pictor v.1.1*** (multi-label annotation) were generated through web-mining from the Google image search database. Single-label classification (on ***Pictor v.1.0*** dataset) and multi-label classifications (on ***Pictor v.1.1*** dataset) were performed to classify common construction objects into three categories, i.e., *building*, *equipment*, and *worker*. In both cases, transfer learning was utilized by pre-training the models with publicly available large datasets. Particularly, VGG-16 (a CNN model) was pre-trained with the ImageNet dataset which contains 1.2 million images encompassing 1,000 different classes. Two fully-connected layers were added to the existing VGG-16's convolutional layers and weights were updated through training and fine-tuning the model with the target dataset (***Pictor v1.0*** and ***Pictor v1.1***). Next, the CNN model was tested on unseen photos.

For single-label classification, the trained model recognized the class labels with 91.1% accuracy (91.9% precision, and 91.1% recall). It was found that most of the misclassified images contained multiple objects. It was also found that the model tends to classify an image as an object that has a larger visual footprint on the image. For multi-label classification, the trained model predicted class labels with 86.0% accuracy (75.5% precision, and 94.9% recall). The reasons for FN detections could be the smaller footprint of some objects and visual occlusion. Also, some high FP rates could be attributed to inaccurate ground-truth labels (i.e., human annotation).

Overall, the performance of the designed models is comparable with the contemporary methods of recognizing construction objects and materials. Models based on transfer learning have achieved remarkable results, even though the target datasets included a limited amount of data. Therefore, the proposed methods can be applied to automatically generate (single-label or multi-label) tags for construction site imagery which would be helpful to readily retrieve particular images based on visual cues. Moreover, the designed models are tremendously fast and require less-intensive computational power. Therefore, these models can be applied to analyze image and video data in real-time, in large-scale, and on mobile devices. In the future, the in-house ***Pictor*** dataset will be expanded to generate more and multiple class labels (e.g., different types of equipment). Also, the annotation algorithm will be further improved to automatically generate natural language-based tags (a.k.a., captions) describing the scene contained within each image. Finally, additional machine learning techniques, such as generative adversarial neural networks (GANs) will be taken into account.

REFERENCES

Alippi C., Disabato S. and Roveri M. (2018). Moving convolutional neural networks to embedded systems: the Alexnet and VGG-16 case. *Proceedings of 17th ACM/IEEE International Conference on Information Processing in Sensor Networks*. IEEE Press. 212-223.

Bottou L. (2010). Large-scale machine learning with stochastic gradient descent. *Proceedings of COMPSTAT*, Springer, 177-186.

Brilakis I., Soibelman L. and Shinagawa Y. (2005). Material-based construction site image retrieval. *Journal of Computing in Civil engineering*, Vol. 19, No. 4, 341-355.

Brilakis I. and Soibelman L. (2008). Shape-based retrieval of construction site photographs. *Journal of Computing in Civil engineering*, Vol. 22, No. 1, 14-20.

Buja A., Stuetzle W. and Shen Y. (2005). Loss functions for binary class probability estimation and classification: Structure and applications. *Working draft*, 1-49.

Chi S. and Caldas C. H. (2011). Automated object identification using optical video cameras on construction sites. *Computer-Aided Civil and Infrastructure Engineering*, Vol. 26, No. 5, 368-380.

Deng J., Dong W., Socher R., Li L.-J., Li K. and Fei-Fei, L. (2009). ImageNet: A large-scale hierarchical image database. *IEEE conference on computer vision and pattern recognition*, 248-255.

Dimitrov A., and Golparvar-Fard M. (2014). Vision-based material recognition for automated monitoring of construction progress and generating building information modeling from unordered site image collections. *Advanced Engineering Informatics*, Vol. 28, No. 1, 37-49.

Ding L., Fang W., Luo H., Love P. E., Zhong B. and Ouyang, X. (2018). A deep hybrid learning model to detect unsafe behavior: integrating convolution neural networks and long short-term memory. *Automation in Construction,* Vol. 86, 118-124.

Fergus R., Fei-Fei L., Perona P. and Zisserman A. (2005). Learning object categories from Google's image search. 1816-1823.

Friedman J., Hastie T. and Tibshirani R. (2001). *The elements of statistical learning* (Vol. 1), New York, Springer.

Ham Y. and Kamari M. (2019). Automated content-based filtering for enhanced vision-based documentation in construction toward exploiting big visual data from drones. *Automation in Construction*, Vol. 105, 102831.

Han K.K. and Golparvar-Fard M. (2017). Potential of big visual data and building information modeling for construction performance analytics: An exploratory study. *Automation in Construction*, Vol. 73, 184-198.

Han K. K. and Golparvar-Fard M. (2015). Appearance-based material classification for monitoring of operation-level construction progress using 4D BIM and site photologs. *Automation in Construction*, Vol. 53, 44-57.

Hinton G. E., Srivastava N., Krizhevsky A., Sutskever I. and Salakhutdinov R. R. (2012). Improving neural networks by preventing co-adaptation of feature detectors. arXiv preprint arXiv:1207.0580.

Kim H., Kim K. and Kim H. (2016). Data-driven scene parsing method for recognizing construction site objects in the whole image. *Automation in Construction*, Vol. 71, 271-282.

Kolar Z., Chen H. and Luo X. (2018). Transfer learning and deep convolutional neural networks for safety guardrail detection in 2D images. *Automation in Construction*, Vol. 89, 58-70.

Krizhevsky A., Sutskever I. and Hinton G. E. (2012). Imagenet classification with deep convolutional neural networks. Paper presented at the Advances in neural information processing systems. *Proceedings of Advances in neural information processing systems*, 1097-1105.

Labelbox. (2019). Labelbox. Retrieved from www.labelbox.com

LeCun Y., Bengio Y. and Hinton G. (2015). Deep learning. *Nature*, Vol. 521, No. 7553, 436.

LeCun Y., Bottou L., Bengio Y. and Haffner P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, Vol. 86, No. 11, 2278-2324.

Luo X., Li H., Cao D., Dai F., Seo J., and Lee S. (2018). Recognizing diverse construction activities in site images via relevance networks of construction-related objects detected by convolutional neural networks. *Journal of Computing in Civil Engineering*, Vol. 32, No. 3, 04018012.

Marinai S., Gori M. and Soda G. (2005). Artificial neural networks for document analysis and recognition. *IEEE transactions on pattern analysis and machine intelligence*, Vol. 27, No. 1, 23-35.

Murphy K. P. (2012). *Machine learning: A probabilistic perspective*, Cambridge, MIT press.

Oquab M., Bottou L., Laptev I. and Sivic J. (2014). Learning and transferring mid-level image representations using convolutional neural networks. *Proceedings of IEEE conference on computer vision and pattern recognition*. 1717-1724.

Perez L. and Wang J. (2017). The effectiveness of data augmentation in image classification using deep learning. arXiv preprint arXiv:1712.04621.

Shin H. C., Roth H. R., Gao M., Lu L., Xu Z., Nogues I., Yao J., Mollura D. and Summers R.M., (2016). Deep convolutional neural networks for computer-aided detection: CNN architectures, dataset characteristics and transfer learning. *IEEE transactions on medical imaging*, Vol. 35, No. 5, 1285-1298.

Siddula M., Dai F., Ye Y. and Fan J. (2016). Unsupervised feature learning for objects of interest detection in cluttered construction roof site images. *Procedia Engineering*, Vol. 145, 428-435.

Simonyan K. and Zisserman A. (2014). Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556.

Son H., Kim C., Hwang N., Kim C. and Kang Y. (2014). Classification of major construction materials in construction environments using ensemble classifiers. *Advanced Engineering Informatics*, Vol. 28, No. 1, 1-10.

Tieleman T. and Hinton G. (2012). Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning*, Vol. 4, No. 2, 26-31.

Wei Y., Liang X., Chen Y., Shen X., Cheng M.-M., Feng J., Zhao Y. and Yan S. (2016). STC: A simple to complex framework for weakly-supervised semantic segmentation. *IEEE transactions on pattern analysis and machine intelligence*, Vol. 39, 2314-2320.

Wu Y., Kim H., Kim C. and Han S. H. (2009). Object recognition in construction-site images using 3D CAD-based filtering. *Journal of Computing in Civil engineering*, Vol. 24, No. 1, 56-64.

Zhang Y., Zhao D., Zhang J., Xiong R. and Gao W. (2011). Interpolation-dependent image downsampling. *IEEE Transactions on Image Processing*, Vol. 20, No. 11, 3291-3296.

Zou J. and Kim H. (2007). Using hue, saturation, and value color space for hydraulic excavator idle time analysis. *Journal of Computing in Civil engineering*, Vol. 21, No. 4, 238-246.