Deep Learning for Identifying Iran's Cultural Heritage Buildings in Need of Conservation Using Image Classification and Grad-CAM*

Mahdi Bahrami
Information Technology Department
Tarbiat Modares University
Tehran, Iran
mahdi_bahrami@modares.ac.ir

Amir Albadvi
Information Technology Department
Tarbiat Modares University
Tehran, Iran
albadvi@modares.ac.ir

March 1, 2023

Abstract

The cultural heritage buildings (CHB), which are part of mankind's history and identity, are in constant danger of damage or in extreme situations total destruction. That being said, it's of utmost importance to preserve them by identifying the existent, or presumptive, defects using novel methods so that renovation processes can be done in a timely manner and with higher accuracy. The main goal of this research is to use new deep learning (DL) methods in the process of preserving CHBs (situated in Iran); a goal that has been neglected especially in developing countries such as Iran, as these countries still preserve their CHBs using manual, and even archaic, methods that need direct human supervision. Having proven their effectiveness and performance when it comes to processing images, the convolutional neural networks (CNN) are a staple in computer vision (CV) literacy and this paper is not exempt. When lacking enough CHB images, training a CNN from scratch would be very difficult and prone to overfitting; that's why we opted to use a technique called transfer learning (TL) in which we used pre-trained ResNet, MobileNet, and Inception networks, for classification. Even more, the Grad-CAM was utilized to localize the defects to some extent. The final results were very favorable based on those of similar research. The final proposed model can pave the way for moving from manual to unmanned CHB conservation, hence an increase in accuracy and a decrease in human-induced errors.

1 Introduction

Two main categories of Cultural Heritage (CH) are tangible and intangible heritages, and the CHBs fall under the former category. The tangible CHs have universal values which must be physically preserved for future generations as an irreplaceable legacy [1, 2]. CHBs are indubitably an integral part of the history and culture of human beings. Throughout the years many of these precious buildings have been in danger of damage due to several reasons, namely material deterioration, natural disasters, presence of visitors, vandalism, etc. [3–5]. Currently, the topic of CH has attracted increasing global attention from scientists and researchers alike, and the scope of its concept is constantly expanding. Most social scientists emphasize on its utility in supporting ethnic and national interests, while many others point to its creative and counter-hegemonic aspects [5, 6].

---

*Preprint .ver 1.0: Under Submission.

1.1 Importance

Endowed with rich CHBs, Iran is ranked 10th in 2022, among all other countries, with 26 UNESCO world heritage sites [7]. Although only 26 of the CHBs in Iran have been registered in UNESCO and not all of them are buildings, the number of CHBs in Iran is of the order of thousands and according to archaeological findings, Iranian architecture dates back to 6,000-8,000 B.C. [8]. One of the reasons why Iran has been unsuccessful in registering more CHBs is the fact that most of these CHBs have not been preserved correctly, if not at all. Even some CHBs are beyond restoration. The CHBs, which fall under the category of immovable tangible CHs, demand more sophisticated methods for conservation since we cannot move them to museums to preserve. Lack of resources in terms of skilled practitioners, budget, and new technologies are just some of the shortcomings that introduce many problems in the conservation process. As regards the usage of state-of-the-art technologies, Iran as a developing country still uses archaic, and sometimes obsolete, manned methods to preserve these precious treasures of humanity. From a broader perspective, many CHBs around the world suffer from such problems as well, so the use of artificial intelligence (AI) techniques such as ML and DL is not a luxury anymore but a necessity. Using ML and DL, we can move toward unmanned conservation of CHB, hence an increase in accuracy and a decrease in human-induced error.

1.2 Research Aim

The aim of this paper was to develop a highly generalized, yet simple, deep learning pipeline for the identification of CHBs in need of preservation, which can be used even in poor countries. We achieved this by making our model as lightweight as possible using a wealth of novel methods, as not all countries have access to expensive resources. This mindset allows for having fewer data and processing power but still reaping satisfying results (Table 3).

1.3 Contribution

Unprecedented in Iran: To the best of our knowledge, and to our surprise, not even a single scientific research had been conducted using ML or DL in the conservation of Iran's CHBs. The body of research outside Iran is not so much either. according to Fiorucci et al. [9] the use of ML in CH literacy has been quite limited in contrast to other fields. We believe that more research in the intersection of AI and CH can change this situation and can pave the way for the prevalence of such techniques in the process of CHB conservation around the world and accrue many benefits to CHB literacy as well.

First-hand Complex Data: We used first-hand data, which had been collected from different sources, as discussed in subsection 3.1. Using first-hand data is important in the sense that not only our experiment would be unprecedented in Iran but globally as well; since no known CHB dataset to date [9] can cover the diversity of types of buildings, types of defects, and color nuances of both Persian and Islamic architecture, like ours.

New combination of Methods: This paper proposes an automated deep learning pipeline for identifying surface damage of CHBs. Having developing countries in mind, we used a combination of state-of-the-art methods to cater to their conservation needs with as little budget as possible. That said, the final deep learning pipeline, using a pre-trained MobileNet, can be run on low-cost devices, for instance a budget mobile phone, to make inference.

- Image classification: define whether a CHB needs preservation or not.
- MobileNet: a very lightweight CNN architecture, but with approximately the same performance as a lot of havier CNNs (e.g., ResNet and/or Inception).
- Grad-CAM: to approximately localize the defects.
- Transfer learning: to reap great results without the need for expensive servers or manpower to take copious images.
- A valid data augmentation pipeline: allows the model to learn more features from the same data.
- Compound regularization method: a combination of four regularization methods together, namely augmentation, dropout, L2 regularization, and batch normalization.

2 Related works

Globally many attempts have been made to use deep learning for damage detection in CHB images. Wang et al. [10] used object detection (OD) with the aid of FasterR-CNN based on a ResNet101 CNN to detect damage in images of masonry buildings with bounding boxes. In another research, Wang et al. [11] used instance segmentation (IS), by the means of a Mask R-CNN model, for damage detection, using a masked colored layer, in glazed tiled CHBs. An

interesting work by Pathak et al. [12] used Faster-RCNN to detect damage in CHBs, but with one major difference to other works. They used point clouds data, instead of images, as the input to their proposed model, and instead rendered point clouds as images which increased the versatility of their model, since capturing photogrammetry doesn't have the same limitations of manually taking photos. Expectedly, damage detection using deep learning is not limited to CHB literacy; for instance, Perez and Tah [13] used OD to detect defects on the images of modern buildings.

As highly revered as OD and IS are, they have some downsides, namely (1) a time-consuming data labeling process with bounding boxes (for OD) or color annotation (for IS); (2) the need for a huge amount of accurately labeled data; (3) detecting only pre-specified types of defects; and (4) much higher computational complexity, in comparison with image classification. This is especially important in the case of developing countries (e.g., Iran), where budgets and resources are limited. That's why despite the prevalence of OD and IS in computer vision, many researchers opted to use the simpler image classification, where each image will be given a label as a whole, and the position of damage is not delineated. As an example, Perez et al. [14] used image classification and CAM layers to classify and localize defects. The downside of their work was not the use of image classification, but using cropped images, which would have been more suitable for object detection rather than image classification.

The usage of image classification and deep learning has not been just for damage detection, but aspects of CHB can benefit from them, as was the case with Llamas et al. [15] who attempted to classify different architectural elements in historical buildings.

In terms of methodology, we followed the footsteps of Llamas et al. [15] and Perez et al. [14] by using image classification over OD and/or IS. Although our work is different in terms of the details of methodology and data. Unlike them, we used data augmentation and a combination of four regularization methods together, which in our case resulted in a 4-5% improvement in metrics (Table 3 and 4).

Research Gap: To the best of our knowledge, most of the works regarding deep learning and CHB use either simplistic data or use the data belonging to a single CHB. As a result, the final trained model lacks the generalization needed to be used for a wide range of buildings in the country of origin. We believe that the data must reflect the variety of real-world data with no editing or cropping. This way the research can come as close as possible to the practical application of using deep learning in the conservation of CHBs. Despite being known as de facto in CV, OD and/or IS need substantial computational resources to process images and detect damage, therefore making these methods infeasible for developing and/or poor countries with so many CHBs (e.g., Iran). Using more lightweight and sophisticated techniques, we can achieve reasonable results but with low-budget and simple devices (e.g., Mobile Phones).

3 Materials and Methods

3.1 Data

For this experiment, we curated a labeled dataset of approximately 10,500 CHB images. In the following, the data curation process is discussed.

3.1.1 Data Collection

The data were gathered from four different sources; (i) The archives of Iran's cultural heritage ministry; (ii) The author's (M.B) personal archives; (iii) images captured on site by the authors (M.B) during the research process and (iv) pictures crawled from the Internet but kept it to a minimum as their distribution differed due to heavy edits and effects. The images that didn't meet the desired quality were removed, to avoid introducing noise to our dataset. Our collected images proved to be very challenging, in the terms of complexity, peculiarity, level of detail, and variation in size, color, characteristics, etc (Figure 1).

Regarding the population of data, as it was infeasible to have access to all the CHBs of Iran, or manually take pictures of them, we tried a random but fair approach to increase the richness of data by taking samples from a wide variety of buildings in terms of architectural style, color theme, quality, time of building, etc. In the process of collecting data different types of criteria were foremost in our minds:

- **Locations**: Semnan, Hamedan, Tehran, Ghazvin, etc.
- **Types**: Mosques, Shrines, Churches, Palaces, etc.;
- **Style**: Islamic, Roman, Persian, etc.;
- **Types**: cracks, deterioration, mold, etc.;
- **Color nuances**: we have images from different times of the day and in different seasons.;

Class/Label	Train set	Dev set	Test set
Class 0			
Class 1			

Figure 1: A few sample images which show the complexity, diversity and variation of our data.

3.1.2 Data cleaning and preprocessing

A number of preprocessing steps were taken before creating our final datasets:

1. Cleaning low-quality images, in terms of relevance, corruption, aspect ratio, grayscale, lighting condition, etc. (Figure A.1).
2. Fixing the auto-rotation EXIF metadata.
3. Finding a good enough resolution and resizing all images to it (i.e., 224x224).
4. Normalizing pixel values to a range of $[-1, 1]$.

3.1.3 Data labeling

Not to exacerbate the existent data imbalance, we chose binary classification over multi-class classification. The negative class (label 0) was used for images that didn't include physical defects and the positive class (label 1) for the ones that did.

Not to become biased in the labeling phase we had three different highly qualified CHB practitioners label the images individually. This way the final label of a single image was determined by the majority vote of these three labelers.

When it comes to labeling, especially image data, we almost always have to deal with some degree of inconsistency, as different practitioners have different experiences, expertise, criteria, etc. To mitigate this effect we defined some criteria by which each labeler had a more consistent and clear guideline to label the images. Figure A.2 shows why it was so crucial to have some criteria that distinctly determine what should be considered a defect (e.g., in terms of length or depth). As regards what types of physical defects were considered in the labeling process, we can enumerate the crack, mold, stain, and deterioration as the most important ones with enough samples in our dataset.

3.1.4 Creating the datasets

After cleaning and preprocessing our data, it was divided into three mutually exclusive and jointly exhaustive sets, namely train, validation (aka dev), and test (Figure 1). To ensure a random but fair division we used stratifying shuffle that's why we have approximately the same ratio between the number images for each label (Table 1).

4

Table 1: The distribution of data; both aggregated and for each dataset separately.

class/label	Total images	Train set	Validation set	Test set
negative/0	1432	1018 (13.8%)	207 (12.99%)	207 (13.28%)
positive/1	9096	6358 (86.2%)	1386 (87.01%)	1352 (86.72%)
Total	10528	7376 (70.06%)	1593 (15.13%)	1559 (14.80%)

As it's evident in Table 1, the notorious yet prevalent problem of data imbalance could be identified. A will be discussed in subsection 4.2 we used a weighted loss function to mitigate this problem by a large margin.

3.2 Convolutional Neural Networks (CNNs)

Synonymous with unassailable performance when it comes to processing image data, the CNNs were a staple in the field of CV since their introduction in 1989 by LeCun et al. [16, 17]. Therefore it was somewhat indubitable that we needed to process our CHB images with this type of NNs to benefit from all the advantages that could accrue to our models by using CNNs. Goodfellow et al. [18] believe CNNs to have three main benefits: translation equivariance, sparse connections, and parameter sharing. A CNN network has less number of learnable parameters in comparison with its conventional fully connected (FC) counterpart. This reduction in the number of parameters is the product of having sparse connections and parameter sharing which enables CNNs to; (i) train faster; (ii) be less prone to overfitting and as results demand fewer train data; and (iii) be able to work with high dimensional data (e.g., images), that their FC counterparts are incapable of. The CNN does the onerous work of feature extraction automatically; the task that without CNNs used to be done by hand engineering the features [19].

In this experiment, we used three of the most prestigious CNN architectures which have shown compelling results and interesting loss convergence, namely ResNet [20], Inception [21], and MobileNet [22].

3.3 Transfer Learning

Dealing with several restraints such as lack of enough data and powerful computers, a methodology called transfer learning was employed to drastically mitigate these impediments. TL tries to transfer the knowledge, a pre-trained model has already learned from a large amount of data, to another model [23]. Generally, TL consists of two main parts. The first part is responsible for customizing the output layer to our problem. The second part fine-tunes the pre-trained model to adapt more to our specific data.

3.4 Class Activation Mapping (CAM)

In spite of the merits of image classification, there is a notorious drawback that lies within, and that is the black-box nature of artificial neural networks (NN). That being said, we don't know whether the model considers pertinent features in an image to decide its class or not. That's why researchers came up with a solution named class activation mapping (CAM) [24].

In this experiment we used gradient-weighted class activation maps (Grad-CAM) [25] which is a CAM method that merges the gradients (aka derivatives) of the final classification, that is the output layer deciding the label of the image, and the output of the final Conv layer of the model to generate a heatmap. The heatmap then is applied to the original image to localize the places that were taken into account when deciding its class/label.

3.5 Regularization

As one of the salient reasons for the occurrence of overfitting is the lack of enough data, which is ubiquitous in CV, we are always in need of more data. Unfortunately getting more brand-new data is not always possible. A workaround is to use the data we already have to increase the number of valid labeled train data, hence a decrease in overfitting as the model is now less capable of naively memorizing the train set [26]. As data augmentation is a staple in CV [26], we almost always opt for using it and this paper is not exempt. Finally, in Figure 2 the result of our proposed data augmentation pipeline after nine runs on the same image can be seen. The data augmentation methods used in this paper can be found in Table 2.

Briefly, to decrease overfitting, which is commonplace in DL models, due to their high capacity in terms of the number of parameters, a combination of four famous methods were used, namely L2 regularization [27], dropout [28], batch normalization layer [29], and data augmentation [26]. The results of this combining approach, as discussed in section 5,

Table 2: The data augmentation methods used in this paper and their corresponding values.

method	value	method	value
random flip	Horizontal	random brightness	0.05
random rotation	0.005	random saturation	0.6 - 1.2
random crop	5%	random contrast	0.75 - 1.1
random quality	80 - 100	random hue	0.03

Figure 2: An example of applying the proposed data augmentation methods on a train image (i.e., nine times). Notice how random, realistic, and valid the augmented versions are.

were quite satisfiable in terms of overfitting and resulted in a very small amount of overfitting (i.e., $< 1\%$) for all of our models.

4 Implementation

4.1 Network Architecture

In the Figure 3 the holistic architecture of our proposed method is represented. Not to process new input images through a data preprocessing pipeline every time, we embedded both the resizing and the normalization preprocessing functions into our network (i.e., pink box). This way, there would be no need to process the unknown images before executing the prediction on them, after the model had been trained.

It was alluded to before that in this experiment we made use of several pre-eminent CNN architectures to tackle the problem at hand and not to be biased toward a certain architecture. As a result, four different networks were implemented, namely ResNet50-v2, ResNet152-v2, InceptionResNet-v2, and MobileNet-v2. One main difference between the ResNet50-v2 and other models is that we trained the ResNet50-v2 from scratch and with randomly initialized weights; while the other three were pre-trained models which were accompanied by TL.

The responsibility of the Global Average Pooling layer (i.e., purple box) was to flatten the output of the last Conv layer into a matrix, which is the desired shape of the input of a fully connected (FC) layer. Before replacing the output of the pre-trained model with a layer of our own, an FC layer (i.e., light blue box) was added to decrease underfitting; the bigger our network becomes the less underfitting we experience, but it also increasing overfitting, that's why a single FC layer proved to provide a desired trade-off, and thus reduced underfitting by a large margin without increasing overfitting too much.

As shown in Figure 3, our model has two outputs. The first (i.e., green box) is responsible for the task of classification, by which each image will be given a label (i.e., negative or positive). The second output on the other hand does the task of localizing the parts by which the model has decided on a certain label for a specific image; this task is done by the Grad-CAM method (i.e., orange box).

Figure 3: The overall architecture of our proposed model/network. Where the values shown in parenthesis below each layer represent the layer's output shape. The N, $n_H^{[L]}$, $n_W^{[L]}$, and $n_C^{[L]}$ refer to the batch size, height, width, and channels of the last layer (L) of the embedded CNN model respectively.

4.2 Evaluation

To evaluate the implemented networks several metrics have been used in an endeavor to meticulously monitor the behavior of the networks at different stages of training. All these metrics will be scrutinized in the following subsections.

4.2.1 Cost function

As mentioned in subsubsection 3.1.4 our two classes were imbalanced and since it would nudge our model to be biased toward the class with more examples (i.e., the positive class), we had to tackle this problem somehow. Having decided in favor of using the class weight method due to its numerous merits the Equation 1 was used to calculate the weight of each class, but it's worth noting that there is a myriad of ways to calculate the weights but as we would fine-tune the calculated weights later on in hyperparameter tuning phase we chose the most widely used:

$$w_c = \frac{n_t}{n_l * n_c} \tag{1}$$

Where w_c, n_t, n_l, and n_c indicate the calculated weight of class c, the total number of images in the dataset, the number of classes, and the number of images in class c respectively. These weights then will be used in the cost function of our networks so that the importance of images belonging to the inferior class outweighs that of the superior class, in a way that network will be rewarded or penalized more when it comes to the images of the class with fewer examples in it. The binary cross-entropy cost function was used, and the way it calculates cost before and after applying class weights can be seen in Equation 2 and 3 respectively. To make it more concrete the first one is used in validation, test, and prediction while the latter is employed in training time; that is we only care about data imbalance during training which is common sense as the network only updates its internal parameters (e.g., weights) in training time and backpropagation.

$$L(\hat{y}, y) = -\bigg(y log(\hat{y}) + (1 - y) log(1 - \hat{y}) \bigg) \tag{2}$$

$$L(\hat{y}, y) = -\bigg((w_1)(y)log(\hat{y}) + (w_0)(1-y)log(1-\hat{y})\bigg) \tag{3}$$

Where y refers to the true label and the $\hat{y}$ to the predicted label of the given record. Note that as we did binary classification and sigmoid activation function for the output layer then $\hat{y}$ is actually the probability ([0, 1]) of the record belonging to the positive class.

4.2.2 Performance measures and metrics

When it comes to the evaluation of our model, several metrics were incorporated to ensure the rigor of our results. As we suffer from imbalanced data the Accuracy can be quite misleading if the model gets biased toward the superior class, so to address this issue four more performance measures were used, namely Precision, Recall, F-Score, and AUC. If anything, the F-Score is the harmonic mean of the Precision and Recall, thus it takes into account both of them to give us a balanced score of the two. Mathematically, Accuracy, Precision and Recall, and F-Score are defined as:

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \tag{4}$$

$$Precision = \frac{TP}{TP + FP} \tag{5}$$

$$Recall = \frac{TP}{TP + FN} \tag{6}$$

$$F-Score = \frac{2 * Precision * Recall}{Precision + Recall} \tag{7}$$

Where TP, TN, FP, and FN are True Positive, True Negative, False Positive, and False Negative respectively. In this paper the FN takes precedence over FP, thus the Recall is more important than precision as the FN is in the denominator of the Recall's Equation 6, however, we tried to balance them as much as possible. The reason is that if an image is falsely labeled as positive then in the worst-case scenario we lose time, but in the case of an image being falsely labeled as negative, then a building in dire need of conservation can be overlooked which might lead to irredeemable destruction. The area under the ROC curve, abbreviated as AUC, was employed in an endeavor to refrain from creating a model biased toward a certain class. AUC demonstrates the power of the model in distinguishing different classes.

5 Results

After slogging through the onerous task of training and fine-tuning the hyperparameters several times, we achieved highly satisfactory results (Table 3). Note that the training process of the ResNet50-v2 doesn't have the fine-tuning step as we trained it from the ground up and with random initial weights. Considering the lack of enough data and computational power, which were alluded to before, it was of no surprise that the networks trained with TL fared the best.

Among the networks that used TL, there is no definite winner, but the MobileNet-v2 had the best performance considering both the performance measures and the computational complexity for both the training and making an inference. That said, MobileNet's lightweight architecture is conducive to training and predicting faster which is especially important for devices with low computational power such as mobile phones, edge devices, etc. which are considered de facto pieces of equipment to monitor CHBs [30].

5.1 Evaluation of MobileNet-v2's Performance

As mentioned before and according to Table 3 the fine-tuned model made with pre-trained MobileNet-v2 was the winner among the other three networks. and its lightweight architecture which is conducive to training and predicting faster is especially important for devices with low computational power such as mobile phones, edge devices, etc. That being said, as the winner among all four network architectures let's scrutinize MobileNet-v2's performance even more. The results of other networks in detail can be found in Figure A.3-A.5. The Table A.1 displays the most important hyperparameters used during the training and fine-tuning of our multiple networks.

Table 3: Final results, after hyperparameter tuning.

Measure	ResNet50V2 [1]			ResNet152V2 [2]			MobileNetV2 [2,3]			InceptionResNetV2 [2]		
	train	**val**	**test**	**train**	**val**	**test**	**train**	**val**	**test**	**train**	**val**	**test**
Loss	0.48	0.47	0.48	0.38	0.38	0.38	0.31	0.32	0.33	0.36	0.36	0.37
Accuracy	0.83	0.84	0.83	0.88	0.89	0.89	0.90	0.90	0.90	0.88	0.88	0.88
Precision	0.87	0.87	0.87	0.92	0.92	0.92	0.95	0.94	0.94	0.91	0.91	0.91
Recall	0.95	0.95	0.95	0.95	0.95	0.96	0.94	0.94	0.94	0.96	0.95	0.96
F-Score	0.91	0.91	0.91	0.93	0.94	0.94	0.94	0.94	0.94	0.93	0.93	0.93
AUC	0.54	0.54	0.54	0.89	0.88	0.88	0.93	0.92	0.90	0.87	0.86	0.85
TP	6040	1310	1287	6056	1319	1296	5961	1311	1274	6082	1319	1295
FP	923	189	192	551	107	114	328	78	76	623	123	135
TN	95	21	22	467	103	100	690	127	139	395	87	79
FN	318	73	67	302	64	302	397	77	79	276	64	59

[1] This NN was trained from scratch and with initial random weights.
[2] These NNs were trained and fine-tuned using TL.
[3] This NN has the best performance among all.

The fine-tuned MobileNet-v2 doesn't suffer from underfitting nor overfitting (Figure 4). As regards the second output of the fine-tuned MobileNet-v2, the localizations seemed completely relevant and attest to the fact that the model had learned the correct features in the train data (Figure 5).

Figure 4: The changes in performance measures reported after each epoch for both the train and validation sets during the training and fine-tuning phase; belonging to the MobileNet-v2 network. the green line indicates the point, in terms of epoch number, where we started to fine-tune some late layers in the pre-trained model.

The output of several Conv layers, aka feature maps, from our fine-tuned MobileNet-v2 network, are visualized in Figure 6; we purposefully chose one layer from the beginning, one from the middle, and another from the end of the network to demonstrate that the more we go deep into the network the more holistic and abstract the detected features will be and vice versa.

6 Discussion

This work demonstrates the facilities of DL in the conservation of CHB by the means of damage detection. As we have collected a diverse set of intricate CHB images, the trained model is very robust and achieved a minimum of 90% for all the metrics we used on the test set. More than our diverse data, using TL, data augmentation, and three different regularization methods in combination, was conducive to reducing overfitting and increasing the generalization power of our model. The salient reasons that attest to why our results are considered to be good enough are (i) Bayes error rate and (ii) the value of performance measures. Although measuring Bayers error rate is a hard and time-consuming task, which was not in the scope of this experiment, we can argue that its value is high, as for instance even a highly skilled

(a) Tarikhane Mosque   **(b)** Bayazid shrine   **(c)** Oljayto porch

(d) Tarikhane (Grad-CAM)   **(e)** Bayazid shrine (Grad-CAM)   **(f)** Oljayto porch (Grad-CAM)

Figure 5: Some samples of the output of Grad-CAM layer of fine-tuned MobileNet-v2 network. The localized defects are shown by a heatmap (from Blue to Red).

Figure 6: A few samples (i.e., 8) of feature maps from the beginning (top), mid-section (middle), and end (bottom) of our fine-tuned MobileNet-v2 network. The input image was the same as that of the Of subfigure c in Figure 5.

CHB practitioner from the south of Iran, would have had a hard time detecting the defects in CHBs from north of the country, considering the peculiarity and idiosyncrasies of each building in our dataset.

According to Mandrekar [31], in the field of CV, values larger than 90% are considered excellent, so it's safe to assume that the MobileNet-v2 had excellent performance, recording values above 90% for all of our metrics. Other than reaching the best performance among other models, the MobileNet-v2 is particularly interesting as it is a faster NN which is particularly important in doing real-time damage detection in devices with low computational resources, such as mobile phones or edge devices. Using our proposed model based on MobileNet-v2 can pave the way for the wide usage of such models in CH sites in Iran and/or around the world with the fewest possible resources.

To compare our results with those of similar researchers, the papers of Llamas et al. [15] and Perez et al. [14] were used, as these were the ones that used image classification, CNN, and TL, just like this experiment. As both of these papers used multiclass classification whereas we used binary classification, we took the average of each metric (e.g., Recall) for all classes, Llamas et al. had ten/10 classes and Perez et al. had four/4 classes; this way we could make their results comparable to those of ours. The comparison of the results on the test set is shown in Table 4.

Table 4: A comparison between the results of similar studies. the reported values are for test set.

	Precision	Recall	F1 Score
Llamas et al. (ResNet) [15]	0.90	0.90	0.90
Perez et al. (VGG-16) [14]	0.90	0.89	0.89
Our fine-tuned model (MobileNet-v2)	**0.94**	**0.94**	**0.94**

The most important challenges and limitations that we faced during this experiment were: (i) needing more data, which is a perennial problem in CV; (ii) lack of suitable computational power; and (iii) inconsistency in labeling due to personal preference and difference in the level of labelers' expertise.

7 Conclusion

This experiment is concerned with applying novel yet matured methods such as DL and CNNs to make the process of conservation of CHBs less prone to errors and more efficient than doing it manually by direct human supervision. By getting Iran's CHB practitioners, the main beneficiaries of this experiment, to use our proposed models besides their old methods, a higher rate of success in detecting physical defects of such buildings can be achieved. We irrevocably believe that CHB practitioners using DL models, such as our proposed one, can identify physical defects more often than either does alone and hopefully as a result, a lower prospect of CHBs deteriorating in structural health.

In an endeavor to practically demonstrate the utilities of DL in CH literature, We developed a fully fledged DL model that classifies the images in need of conservation and even more approximately localizes the defects to help the CH practitioners identify defects in a timely manner, and as a result speed of the process of CHB conservation as well as increasing its accuracy. In spite of all the limitations, we achieved very good results with a score of at least 94% for Precision, Recall, and F1-Score, which were about 4-5% more than similar works (Table 4).

As regards future works, addressing the limitations we faced can open up a plethora of opportunities in terms of methods and outputs. for instance, if had access to a large amount of labeled data and powerful servers, physical or in the cloud, then object detection or instance segmentation would be more useful and could elicit more accurate and user-friendly results from our data. Having gotten traction in the past few years, the generative adversarial networks (GANs) can be utilized in our network architecture to propose restoration based on the label and localizations our proposed model offers.

References

[1] F. J. Lopez, P. M. Lerones, J. Llamas, J. Gomez-Garcia-Bermejo, E. Zalama, A review of heritage building information modeling (h-bim), Multimodal Technologies and Interaction 2 (2018). doi:doi:10.3390/mti2020021.

[2] M. Vecco, A definition of cultural heritage: From the tangible to the intangible, Journal of Cultural Heritage 11 (2010) 321–324. doi:doi:https://doi.org/10.1016/j.culher.2010.01.006.

[3] Y. Chen, G. Medioni, Object modeling by registration of multiple range images, volume 3, Publ by IEEE, 1991, pp. 2724–2729. doi:doi:10.1109/robot.1991.132043.

[4] J. Markiewicz, S. Lapinski, P. Kot, A. Tobiasz, M. Muradov, J. Nikel, A. Shaw, A. Al-Shamma'a, The quality assessment of different geolocalisation methods for a sensor system to monitor structural health of monumental objects, Sensors (Switzerland) 20 (2020) 2915. doi:doi:10.3390/s20102915.

[5] F. Stanco, S. Battiato, G. Gallo, Digital Imaging for Cultural Heritage Preservation: Analysis, Restoration, and Reconstruction of Ancient Artworks, 2011. URL: https://books.google.com/books?id=QHnBxQ2xhGQC&pgis=1.

[6] C. Brumann, Cultural Heritage, Elsevier Inc., 2015. doi:doi:10.1016/B978-0-08-097086-8.12185-3.

[7] U. W. H. Centre, Iran (islamic republic of) heritage sites, 2022. URL: https://whc.unesco.org/en/statesparties/IR/.

[8] M. Hejazi, B. Hejazi, S. Hejazi, Evolution of persian traditional architecture through the history, Journal of Architecture and Urbanism 39 (2015) 188–207. doi:doi:10.3846/20297955.2015.1088415.

[9] M. Fiorucci, M. Khoroshiltseva, M. Pontil, A. Traviglia, A. D. Bue, S. James, Machine learning for cultural heritage: A survey, Pattern Recognition Letters 133 (2020) 102–108. doi:doi:10.1016/j.patrec.2020.02.017.

[10] N. Wang, X. Zhao, P. Zhao, Y. Zhang, Z. Zou, J. Ou, Automatic damage detection of historic masonry buildings based on mobile deep learning, Automation in Construction 103 (2019) 53–66. doi:doi:10.1016/j.autcon.2019.03.003.

[11] N. Wang, X. Zhao, Z. Zou, P. Zhao, F. Qi, Autonomous damage segmentation and measurement of glazed tiles in historic buildings via deep learning, Computer-Aided Civil and Infrastructure Engineering 35 (2020) 277–291. doi:doi:10.1111/mice.12488.

[12] R. Pathak, A. Saini, A. Wadhwa, H. Sharma, D. Sangwan, An object detection approach for detecting damages in heritage sites using 3-d point clouds and 2-d visual data, Journal of Cultural Heritage 48 (2021) 74–82. doi:doi:10.1016/J.CULHER.2021.01.002.

[13] H. Perez, J. H. M. Tah, Deep learning smartphone application for real-time detection of defects in buildings, Structural Control and Health Monitoring 28 (2021) e2751. doi:doi:https://doi.org/10.1002/stc.2751.

[14] H. Perez, J. H. M. Tah, A. Mosavi, Deep learning for detecting building defects using convolutional neural networks, Sensors 19 (2019) 3556. doi:doi:10.3390/s19163556.

[15] J. Llamas, P. M. Lerones, R. Medina, E. Zalama, J. Gomez-Garcia-Bermejo, Classification of architectural heritage images using deep learning techniques, Applied Sciences (Switzerland) 7 (2017) 992. doi:doi:10.3390/app7100992.

[16] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, L. D. Jackel, Backpropagation applied to handwritten zip code recognition, Neural Computation 1 (1989) 541–551. doi:doi:10.1162/NECO.1989.1.4.541.

[17] W. Fang, P. E. Love, H. Luo, L. Ding, Computer vision for behaviour-based safety in construction: A review and future directions, Advanced Engineering Informatics 43 (2020) 100980.

[18] I. Goodfellow, Y. Bengio, A. Courville, Deep learning, MIT press, 2016.

[19] J. Gu, Z. Wang, J. Kuen, L. Ma, A. Shahroudy, B. Shuai, T. Liu, X. Wang, G. Wang, J. Cai, et al., Recent advances in convolutional neural networks, Pattern recognition 77 (2018) 354–377.

[20] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition 2016-December (2015) 770–778. doi:doi:10.48550/arxiv.1512.03385.

[21] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, Going deeper with convolutions, Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition 07-12-June-2015 (2014) 1–9. doi:doi:10.48550/arxiv.1409.4842.

[22] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, H. Adam, Mobilenets: Efficient convolutional neural networks for mobile vision applications (2017). doi:doi:10.48550/arxiv.1704.04861.

[23] F. Zhuang, Z. Qi, K. Duan, D. Xi, Y. Zhu, H. Zhu, H. Xiong, Q. He, A comprehensive survey on transfer learning, Proceedings of the IEEE 109 (2019) 43–76. doi:doi:10.48550/arxiv.1911.02685.

[24] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, A. Torralba, Learning deep features for discriminative localization, Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition 2016-December (2015) 2921–2929. URL: http://arxiv.org/abs/1512.04150.

[25] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, D. Batra, Grad-cam: Visual explanations from deep networks via gradient-based localization, International Journal of Computer Vision 128 (2016) 336–359. doi:doi:10.1007/s11263-019-01228-7.

[26] K. Maharana, S. Mondal, B. Nemade, A review: Data pre-processing and data augmentation techniques, Global Transitions Proceedings 3 (2022) 91–99. doi:doi:10.1016/J.GLTP.2022.04.020.

[27] C. Cortes, M. Mohri, A. Rostamizadeh, L2 regularization for learning kernels, Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence, UAI 2009 (2012) 109–116. doi:doi:10.48550/arxiv.1205.2653.

[28] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, R. R. Salakhutdinov, Improving neural networks by preventing co-adaptation of feature detectors (2012). doi:doi:10.48550/arxiv.1207.0580.

[29] S. Ioffe, C. Szegedy, Batch normalization: Accelerating deep network training by reducing internal covariate shift, 32nd International Conference on Machine Learning, ICML 2015 1 (2015) 448–456. doi:doi:10.48550/arxiv.1502.03167.

[30] M. Maksimovic, M. Cosovic, Preservation of cultural heritage sites using iot, 2019 18th International Symposium INFOTEH-JAHORINA, INFOTEH 2019 - Proceedings (2019). doi:doi:10.1109/INFOTEH.2019.8717658.

[31] J. N. Mandrekar, Receiver operating characteristic curve in diagnostic test assessment, Journal of Thoracic Oncology 5 (2010) 1315–1316. doi:doi:10.1097/JTO.0B013E3181EC173D.

A   Appendix: Supplementary Materials

(a) elongated

(b) irrelevant to CHB

(c) too little light

(d) blurry

(e) scans from printed images

(f) too much light

Figure A.1: Some examples of unsuitable images, which were omitted in the data preprocessing phase.

Figure A.2: Comparing a picture with small-sized defects (left) with a picture with large-sized defects (right). The defects are delineated in red, for more clarity.

Figure A.3: The changes in performance measures reported after each epoch for both the train and validation error for the ResNet50-v2 network.

Figure A.4: The changes in performance measures reported after each epoch for both the train and validation error belonging to the ResNet152-v2 network.

Figure A.5: The changes in performance measures reported after each epoch for both the train and validation error belonging to the InceptionResNet-v2 network.

Table A.1: The salient hyperparameters used to train our networks.

hyperparameter	ResNet50V2	ResNet152V2	MobileNetV2	InceptionResNetV2	default
batch_size	32	32	32	32	-
feature range	[-1, 1]	[-1, 1]	[-1, 1]	[-1, 1]	[0, 255]
dropout	0.8	0.7	0.5	0.5	0.0
L2 lambda	0.1	0.01	0.01	0.01	0.01
class weight	{'0': 3.5, '1': 1}	{'0': 2, '1': 1}	{'0': 2, '1': 1}	{'0': 3, '1': 1}	{'0': 1, '1': 1}
optimizer	Adam	Adam	Adam	Adam	RMSprop
first step [1]					
learning rate	0.01	0.005	0.001	0.001	0.001
decay steps	1,000	1,000	1,000	1,000	100,000
decay rate	0.96	0.96	0.96	0.96	0.96
#epochs	100	10	30	30	-
second step [2]					
learning rate	-	1e-8	1e-6	1e-8	0.001
decay steps	-	300	300	300	100,000
decay rate	-	0.96	0.96	0.96	0.96
#epochs	-	10	10	10	-
#unlocked layers [3]	-	64 out of 564	54 out of 154	80 out of 780	-

[1] Refers to the first part of TL where we train the weights of the added FC layers at the end of pre-trained models.
[2] Refers to the second part of TL where we fine-tune the weights of several filters in the pre-trained model.
[3] This value shows how many of the later layers in the pre-trained models were unlocked to be fine-tuned in second step of TL.