SYNTHETIC DATA GENERATION FOR INTELLIGENT INSPECTION OF STRUCTURAL ENVIRONMENTS

NOSHIN HABIB

Doctoral Program in Mechanical Engineering

APPROVED:

Angel Flores-Abad, Ph.D., Chair

Ahsan Choudhuri, Ph.D., Co-Chair

Joel Quintana, Ph.D.

Virgilio Gonzalez, Ph.D.

Stephen Crites, Ph.D.
Dean of the Graduate School

To my wonderful parents, whose kindness, support, and constant prayers have raised me to become who I am.

To my incredible brother, whose friendship and encouragement have inspired me to persevere.

To my beloved other half, whose unconditional love and acceptance have taught me the virtue of patience and positivity.

To my amazing best friend, whose constant hours of conversation and laughter have encouraged me to always be my authentic self.

"Verily with hardship comes ease" (94:5)

SYNTHETIC DATA GENERATION FOR INTELLIGENT INSPECTION OF STRUCTURAL ENVIRONMENTS

by

NOSHIN HABIB

DISSERTATION

Presented to the Faculty of the Graduate School of

The University of Texas at El Paso

in Partial Fulfillment

of the Requirements

for the Degree of

DOCTOR OF PHILOSOPHY

Doctoral Program in Mechanical Engineering

THE UNIVERSITY OF TEXAS AT EL PASO

December 2022

Acknowledgements

Throughout the journey from my Undergraduate studies to the completion of my Ph.D., UTEP has been my home. The wonderful professors and peers I have met during my time here are ones that I cannot express my gratitude for in words. I would like to specifically acknowledge the support of the UTEP Aerospace Center for giving me the opportunity to conduct this research.

Thank you to my wonderful teammates Mousumi Rizia, Guillermo Ortega, and Julio Reyes for their constant support and guidance. Thank you as well to my committee members Angel Flores-Abad, Ahsan Choudhuri, Joel Quintana, and Virgilio Gonzalez for their patience and mentorship throughout this process.

And finally, thank you to my family and friends for keeping me sane in the midst of some of the most difficult years of my life. Without you all, I would not have completed this degree.

This research is supported by the US Department of Energy, under award DoE Award Number: DEFE0031655.

Abstract

Automated detection of cracks and corrosion in pavements and industrial settings is essential to a cost-effective approach to maintenance. Deep learning has paved the path for vast levels of improvement in the area. Such models require a plethora of data with accurate ground truth and enough variation for the model to generalize to the data, which is not widely available. There has been recent progress in computer graphics being used for the creation of synthetic data to address the issue of deficient data availability, but it is limited to specific objects, such as cars and human beings. Textures and deformities within such objects are left unexplored.

This study introduces an approach to synthetically produce a dataset of pavement images with cracks and a dataset of industrial images with corrosion using Unreal Engine 5, a 3D Computer Graphics Gaming Engine. For both datasets, a novel annotation technique is used to provide labels with pixel-level detail. For the feasibility of use with object detection algorithms, a python code is created for bounding box derivation of the segmented ground-truth. The aim of the datasets is not to fully replace a real dataset altogether, but to save the time and resources that would be required to gather enough images with high levels of variety, without the need for manual annotation.

The virtual datasets are trained in combination with real data and are evaluated using the deep learning framework You Only Look Once (YOLOv4). The datasets are also tested on real data to show the transferability of learning from synthetic data to real-world applications. The datasets will be publicly available so that they can easily be altered for the needs of the user. This work provides evidence suggesting that (i) the creation of publicly available synthetic data using open-source gaming engines does not have to be limited to large objects and can significantly cut down on time and resources needed for accurately labeled data, and (ii) training on virtual data improves performance on detecting cracks and corrosion.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

Inspection of structural environments is a crucial component of safety and maintenance. If fractures are not detected early on, this can prove to be detrimental to workers and accrue high costs for repair. Robotic technologies and deep learning methodologies for the use of such inspection have become a growing area of interest. Such technologies help reduce risk to human operators and in reducing costs due to the downtime of equipment. In the energy sector, high interest is placed on power plant components, where safe operation is paramount. Likewise, in the public sector, high interest is placed in pavement roads, where conditions directly affect civilians.

This work is part of an overarching project to create an intelligent inspection system using an unmanned aerial vehicle (UAV) in GPS-denied environments. One of the key components of the project is the ability to detect cracks and corrosion in such environments, which is the focus of this work. Deep learning has paved the path for vast levels of improvement in the area of automated inspection. However, such models require a plethora of data with accurate ground truth, which is not widely available.

To combat this issue, a novel framework is proposed to create additional training data for such deep learning models. Synthetic data has gained much traction in recent years due to the exponential growth in computer graphics technology, but has not been used in the realm of structural environments. Through the methodology proposed in this work, the data deficiency that is prevalent for both cracks and corrosion will be mitigated.

This chapter will introduce the integration of three emerging concepts in the fields of artificial intelligence, computer vision, and industrial environments: Deep learning, synthetic data, and intelligent inspection. Each concept is inspected by its impact and its

development over time.

1.1 Problem Statement

Using robust learning methods for structural inspection requires a high level of data that is difficult to attain without the necessary funds and equipment. Furthermore, the quality of the data is equally as important as the amount of data present. Synthetically generated data is widely being used in computer vision to lower costs and provide greater flexibility with limitless quantity. At the time of this publication, this method has not been implemented for textures on pavements and structural environments in the scale that this work attempts to accomplish. This study creates synthetically rendered images using Unreal Engine 5 (UE5), an open-source gaming engine [21] for various object detection tasks.

1.2 Objective

Most synthetic data has been limited to objects such as those present in indoor scenes [42]. Although some work has been done on synthesizing texture-based fractures such as cracks [24], at the time of this publication, this method has not been implemented for cracks and corrosion in various environments. The novelty of this study is to synthetically generate textures for object detection to be used with other datasets to increase the size and variety of the training model. The following are the main contributions of this project:

1. Creation of a novel end-to-end framework for synthetic data generation of structural environments with texture-level fractures.

2. Creation of a new synthetic dataset for pavement cracks to minimize time and increase the variety required for additional data.

3. Creation of a new synthetic dataset for crevice, pitting, and concrete corrosion related to power plants to address the lack of datasets publicly available.

4. Generation of an automatic ground-truth method for pixel-level detail and bounding box
   derivation.

5. Implementation of deep learning frameworks for evaluation to show the value added by the synthetic dataset generation.

1.3 Research Significance

This work provides the first step towards a robust synthetic data generation system to combat the issues of not having enough data for the purposes of crack and corrosion detection using deep learning. This work also proposes a new method for automated annotation of texture-level faults within structural environments. Two novel synthetic datasets are created using a state-of-the-art 3D graphics game engine, UE5. Due to the feasibility of the proposed framework, the datasets can be further altered based on the needs of the user. The proposed method can mitigate the need for generating large amounts of real data that require ample time for annotation with the risk of human error. Additionally, the collection of such data can be costly, time-consuming, and dangerous. Synthetic data allows for the elimination of these issues, along with the ability to create instant changes as required. The framework presented in this work can be used for various domain applications and is not limited to the ones shown.

1.4 Deep Learning

Artificial intelligence (AI) is a rapidly advancing field, containing a plethora of research areas within it. Machine learning is a subset of AI, which focuses on the engineering implementation of AI. Machine learning includes a multitude of methods, of which artificial neural networks (ANN) are one of them. The relationship between these concepts can be seen in Figure 1.1. ANNs, first proposed by [50], are an assembly of interconnected nodes,

loosely based on the functionality of neurons in human beings. The capability of the network is stored in weights. Neural networks learn from a set of patterns that are used to train the model [26]. The term "network" refers to the system of artificial neurons, which can range in complexity. In supervised learning, the network is presented with an input pattern and the network's response is compared with the target output. The weights of the nodes are altered based on the performance of the model. This sequence is repeated until it has converged, or has reached a certain number of epochs.

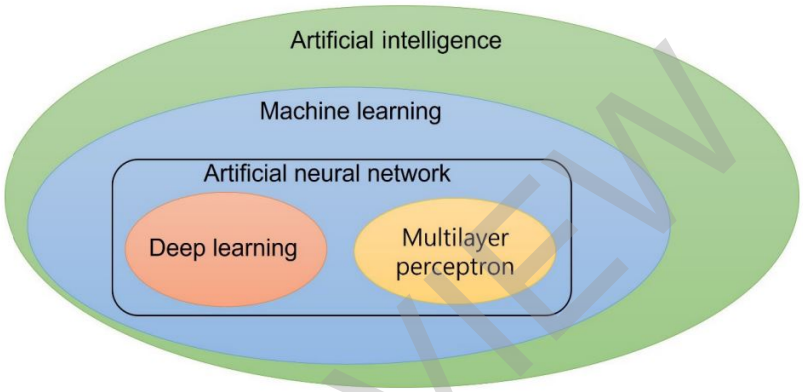

Figure 1.1: Relationship between artificial intelligence, machine learning, neural networks, and the different types of networks [65].

Rosenblatt [63] proposed a "multilayer perceptron" to mimic the signal transduction process of human nerve cells. Deep learning neural networks were first used in [28], where a multilayer perceptron was created with multiple hidden layers. Deep learning is the term used when this type of neural network architecture is implemented in a task. The difference between deep learning networks and regular neural networks can be seen in Figure 1.2.

With the advances in deep learning technology came the concept of convolutional neural networks (CNN), which are primarily used for image data. A CNN is an image pattern recognition algorithm that facilitates the end-to-end process from feature extraction to classification. They can also be applied to segmentation and object detection tasks. CNNs contain convolution, pooling, and fully-connected dense layers, as shown in Figure 1.3.

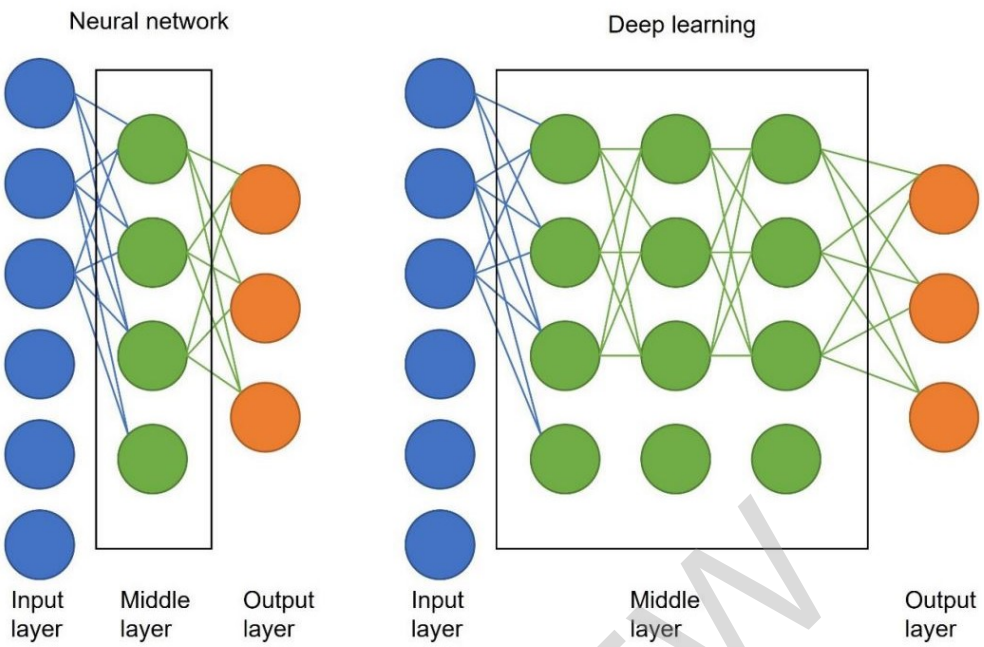

Figure 1.2: Difference between traditional neural networks and deep learning [65].

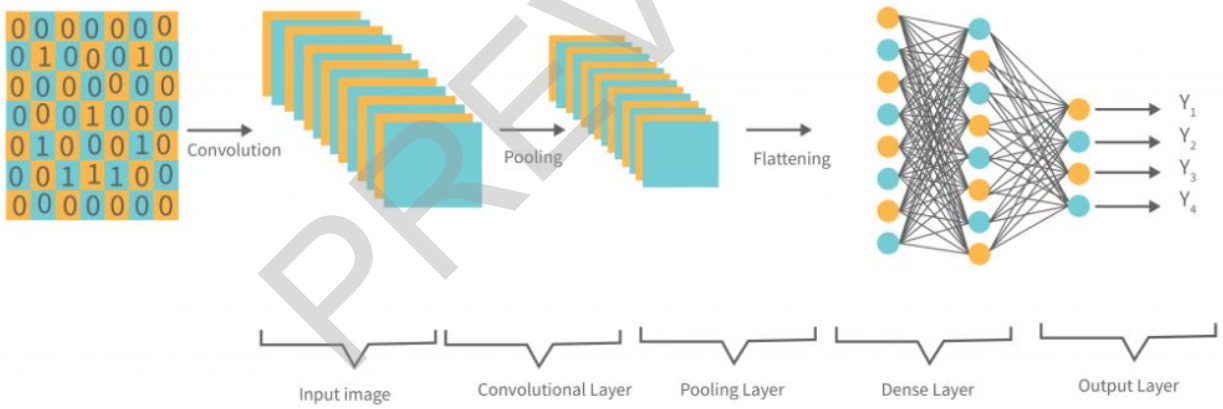

Figure 1.3: Typical CNN architecture [31].

The convolutional layer comprises of multiple feature maps, each of which contains various neurons. The convolutional kernel allows for each neuron to connect to a local region of the former feature surface. These feature surfaces allow the convolutional layer to identify the output of the neurons. The neural network uses convolutional operations

to extract features of the input data. The pooling layer aims to further modify the layer output and is typically conducted after the convolutional layer. This layer down-samples along the spatial dimension to decrease the number of parameters. Typically, the maximum output is reported within a region. After pooling, dense fully-connected layers are applied to flatten the feature into a one-dimensional arrangement. Finally, the output value of the last dense layer is forwarded to an output layer, which performs the classification, detection, or segmentation.

1.4.1 Training, Validation, and Testing

The method of training that will be used for this project is the 3-set split. First, a training set is used to train the model to learn the necessary features and generalize on the data. The validation set is used during training to ensure that the model is not overfitting on the training data. This allows for hyperparameter tuning to help the model generalize better. The testing set is the final set that the data is tested on, which should contain data that has never been seen by the model before. The true error measured for the model will be from the testing set, while the validation set is to ensure the model has been trained adequately. Typically, the training and validation set should be split between 80/20 and 90/10 respectively [78].

1.5 Synthetic Data

1.5.1 Methods for Mitigating Small Amounts of Data

There are various common approaches to reusing trained networks for additional purposes. There are also methods to fabricate an increased amount of data available for training without gathering or annotating more images. This section will discuss a few of these methods prior to delving into synthetic data created using gaming engines.

Transfer Learning and Fine Tuning

Transfer learning and fine-tuning are techniques in which trained weights from one application are transferred to another. For example, a network trained for autonomous driving in urban areas can be applied to autonomous rescue missions in rural environments. Both methods make use of the fact that the last layer of a neural network is responsible for the classification or detection task. This layer also indicates the number of classes the network can identify.

Transfer learning replaces the last layer with a layer containing the correct number of classes. Then, the correct dataset for that domain is used for training. The other layers of the network are unchanged, and take advantage of the features learned by the previous training.

For fine-tuning, the previous layers are subject to change. The custom dataset in the last layer makes special adjustments to the previous layers based on the new findings. When using this method, a smaller learning rate is preferred. This is because the shallower layers learn lower-level features. Deeper layers of the network differ more across domains and can sustain a higher rate of change. However, if shallow layers are changed too frequently, the network is subject to overfitting.

Neural network techniques used to train networks from the ground up can be adapted for transfer learning. A shallow network that corresponds to the last layers of the pre-trained network being targeted can be trained from scratch for the specific dataset being applied. The last layers can then be replaced by the shallow network with its corresponding weights. Lastly, the network can be fine-tuned after more training on the dataset. An example of this methodology can be found in [73], where a shallow version of a VGG network architecture was created and trained before being extended.

Pre-trained Networks Deep learning frameworks and neural network architectures can be trained on large datasets such as ImageNet [13], Pascal VOC [18], or COCO [43]. These weights are then made publicly available for use prior to training using other datasets.

Due to the large volume of data the pre-trained weights have been trained on, the neural network can abstract generalized feature representations for various object classes. Then, customized feature extraction can be created when adapting the neural network training to the desired dataset.

Using a pre-trained network for a custom application can significantly improve the accuracy of the model, even if the custom dataset differs greatly from the large dataset the network was pre-trained on. For example, Eitel et al. [16] and Schwarz et al. [69] trained a network of range image data by using the pre-trained weights from ImageNet, which provide RGB images. In another example, [17] also uses a network pre-trained on ImageNet for skin cancer classification. Evidently, a network pre-trained on the same large dataset can be used for completely different applications and is one of the simplest methods to reduce necessary training data.

Data Augmentation

Image data augmentation consists of various techniques. The different types of data augmentation types can be seen in Figure 1.4. Data augmentation aims to increase the size of the data through manipulation, without the need for collecting additional data. Data augmentation is typically done during training to help the model generalize its learning and avoid overfitting.

The most common image augmentation techniques are basic image manipulation. Kernel filters are used in image processing to blur or sharpen the image by adding a gaussian blur filter or high contrast edge filter respectively. Geometric transformations include augmentations such as flipping, rotating, cropping, and translating. Random erasing randomly selects a patch of the image and masks the area with a certain value to help combat image recognition with occlusion. Mixing images also aims to combat occlusion by taking random samples from other images in the dataset and adding them to a patch of the current image. Finally, color space transformations include changing the hues, brightness, and contrast of the image to add complexity to the training data.