Contract No:

This document was prepared in conjunction with work accomplished under Contract No. 89303321CEM000080 with the U.S. Department of Energy (DOE) Office of Environmental Management (EM).

A U.S. DEPARTMENT OF ENERGY NATIONAL LAB • SAVANNAH RIVER SITE • AIKEN, SC • USA

Image Classification Using Convolutional Neural Networks to Automate Visual Inspection of CCO Containers

Isaac Garon

August 2023

SRNL-STI- 2023-00408, Revision 1

DISCLAIMER

Printed in the United States of America

Prepared for
U.S. Department of Energy

Image Classification Using Convolutional Neural Networks to Automate Visual Inspection of CCO Containers

Abstract. The Savannah River Site is automating the receipt and inspection of Criticality Control Overpack drums to aid in nuclear waste disposal. The Advanced Engineering group is automating this receipt and inspection process to assist the site's goals. Part of this process includes visually inspecting CCO drums to ensure that no defects or security risks are present. This research focuses on developing a machine learning model to automatically classify drums as passing or failing inspection. The machine learning model was implemented using the open-source Tensorflow library and uses a Convolutional Neural Network architecture to extract features and differentiate between images. The model was able to achieve an average of 85% accuracy on a dataset of 288 drums. This project's goal was to prove the validity of computer vision in an automation process and provide a starting place for continued research.

Introduction

Waste Management Automation

The Advanced Engineering group at the Savannah River National Laboratory (SRNL) was tasked with automating the receipt and inspection of Criticality Control Overpack (CCO) containers to aid in nuclear waste management. The Savannah River Site (SRS) plans on removing excess nuclear waste by storing neutralized nuclear material in CCO drums. To ensure the safety of this process, CCO containers are inspected upon receipt to ensure that design specifications are met and no security risks are found. After receiving the CCO drums from the manufacturer, an automated process will unpack and inspect these drums by scanning the outside of the containers for any dents, holes, or other defects using a laser scanner. In addition to ensuring a standard of quality, safeguarding against security breaches in a receiving process is critical; therefore, the inside of these drums must also be visually inspected to ensure there is no internal damage or foreign objects inside the drums. While laser profilers are excellent at mapping surfaces, large open spaces require different techniques for validation. This automation process sought to automate visual inspection of CCO drums using a machine learning image classification model that could differentiate between drums without foreign items and drums containing foreign items.

Image Classification using Machine Learning

Computer vision is the discipline of training computers to recognize images as humans do, and it encompasses many different applications. One such application is image classification, the process of differentiating images by assigning a label to an image from predetermined classes. For example, an image classification algorithm could determine if a picture is an image of a cat or an image of a dog; it labels the entire image as belonging to the “cat” class or the “dog” class whether a cat or a dog is present in the image. This is an example of a binary image classification problem as it discriminates images into two distinct categories. Similarly, binary classification can be applied to CCO drums. If drums are split into two categories: “pass” and “fail”, then a binary image classification algorithm can automatically determine if a drum passes or fails the inspection. This process is a great application of binary image classification since the images generated by the automation process will be consistent; furthermore, machine learning models are robust and can handle many different scenarios by adapting to unseen images. Since this technology has not been implemented in an automation process at SRNL before, the scope of this research was to explore the viability of machine learning in automation by testing a binary image classification machine learning algorithm on images of CCO drums. The goal of this project was to understand computer vision systems, explore various machine learning algorithms, and implement a binary image classification machine learning algorithm. The results of this project were reviewed to determine the viability and benefits of continuing research on computer vision to aid in automation.

Machine Learning and Computer Vision

Machine Learning

Machine learning is the term for programs that learn and adapt to solve a problem without following explicit instructions; in other words, machine learning programs model solutions to problems by recognizing patterns in datasets that it is trained on. A specific implementation of machine learning is called a model as it models the given problem and produces a solution. For example, a common use of machine learning is handwriting recognition. Machine learning models are trained on many examples of handwritten letters and numbers; after inferring patterns of handwriting from this training data, the model is then able to correctly identify letters and numbers on handwriting examples it has not seen before. In this case, the machine learning algorithm solves the problem of recognizing handwriting by inferring patterns from a dataset of handwritten examples and predicts the output of a new handwritten example. Since machine learning is such a broad topic, there are numerous types of machine learning algorithms used for every application. This research focuses on functional algorithms which map an input to an output.

Machine learning models operate by using weights to determine the significance of certain patterns. Essentially, a machine learning model will predict outputs from the training data; as the model continues training on a dataset, it will tune these weights to identify key patterns for the current problem. There are several types of machine learning training; however, this research focuses on supervised algorithms. Supervised algorithms train the model on a dataset where the labels are given and whether the model's predictions are right or wrong modifies the weights of the algorithm. For example, imagine a model predicts an output because of a certain recognized pattern. If the output is correct, then the model will increase the weight given to that pattern. In future predictions, if that same pattern occurs, then the model will be more likely to predict the answer that was previously correct. This process of supervised training forms the basis of machine learning algorithms used on images and videos.

The pipeline for machine learning applications can be simplified into three categories: data preparation, model training, and model deployment. Data preparation involves gathering a quality dataset and processing datapoints to reduce noise and error. As previously discussed, model training involves propagating data through the model while the model continuously adjusts weights to learn the problem and generalize a solution. Deploying the model is the final step where the finished model is used to make predictions on new data outside of the training set. This process is fundamental to any machine learning application.

Computer Vision and Neural Networks

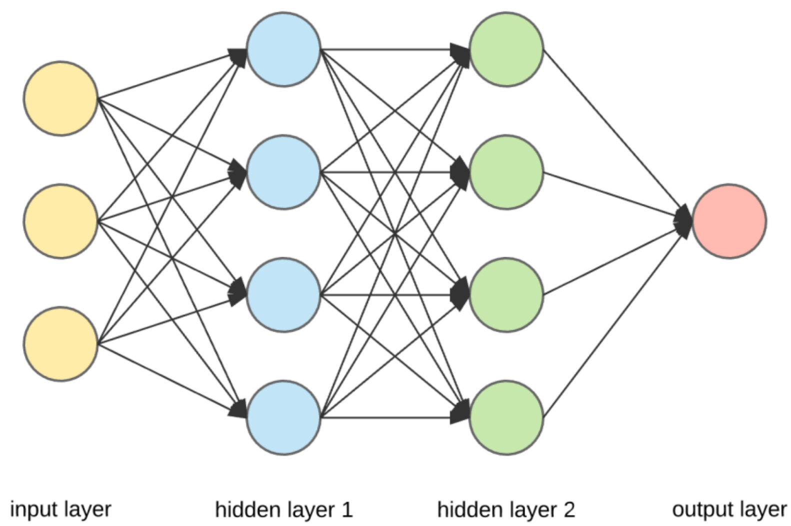

Figure 1. Neural Network Example

Computer vision, specifically, is the application of machine learning models to images and videos. Machine learning models can analyze images for patterns and features to mimic human vision and perception. While computer vision encompasses many types of algorithms, the most common architecture is a Convolutional Neural Network (CNN). A CNN is a type of neural network machine learning algorithm. Neural network models mimic human neural architecture by using connected layers of neurons to identify patterns and features. While human neurons are complex biological structures, a computer neuron is simply a function where an input is mapped to an output. In neural networks, vast layers of neurons are connected to extract meaning from images. As an image is propagated through the model, it will pass through each layer where meaningful information is extracted using functions defined in the neurons; usually, these functions are statistical equations. There are numerous types of neural network layers – each with their own application. One such layer is a convolutional layer which is the foundation of a Convolutional Neural Network.

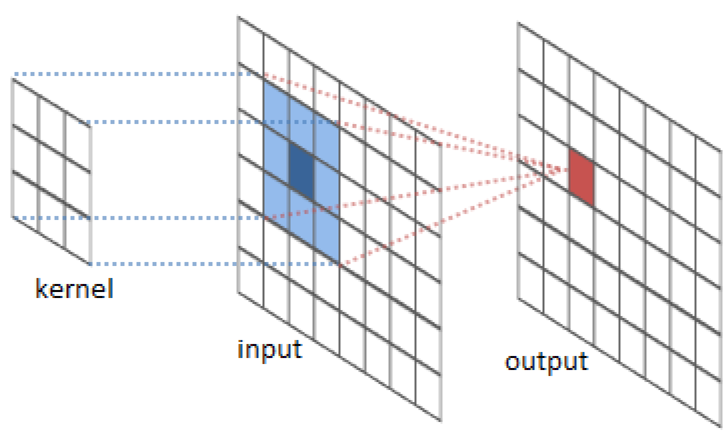

Figure 2. Representation of a Convolutional Layer

Convolutions are an operation performed between two matrices. Images are digitally represented using 2D matrices; each pixel of an image corresponds to a value within the matrix. To perform a convolution, a small kernel matrix "slides" across the larger image matrix. As the kernel slides across the image, the dot product of these two matrices is performed. This process repeats until the kernel reaches the end of the image matrix. Next, a pooling layer is used to compress the resulting matrix. Because the resulting convolution matrix is quite large, the pooling layer takes a representative pixel from a small region to create a new resulting matrix. This pixel value can be the average or the maximum of the pixel values. For example, a max pooling layer will take the maximum value of 4 adjacent indices and place this in the corresponding index of a new matrix. This process will continue for the entire matrix resulting in a compressed matrix that represents the output of the convolutional layer. The convolutional and pooling layers extract features from images to be analyzed by the model. In order to make sense of these features, dense layers perform functions on these feature maps to classify the image. These dense layers are made up of many neurons whose inputs are connected to the outputs of other neurons. These final layers are often called fully connected layers since the inputs of all neurons are connected to the output of the previous layer. In summary, a Convolutional Neural Network uses convolutional and pooling layers to extract important features from the image while fully connected dense layers use weights to classify the image into two or more categories. Less complex problems require less layers while more complex problems require more layers; however, all CNN models need at least one Convolutional layer and one dense layer to classify images.

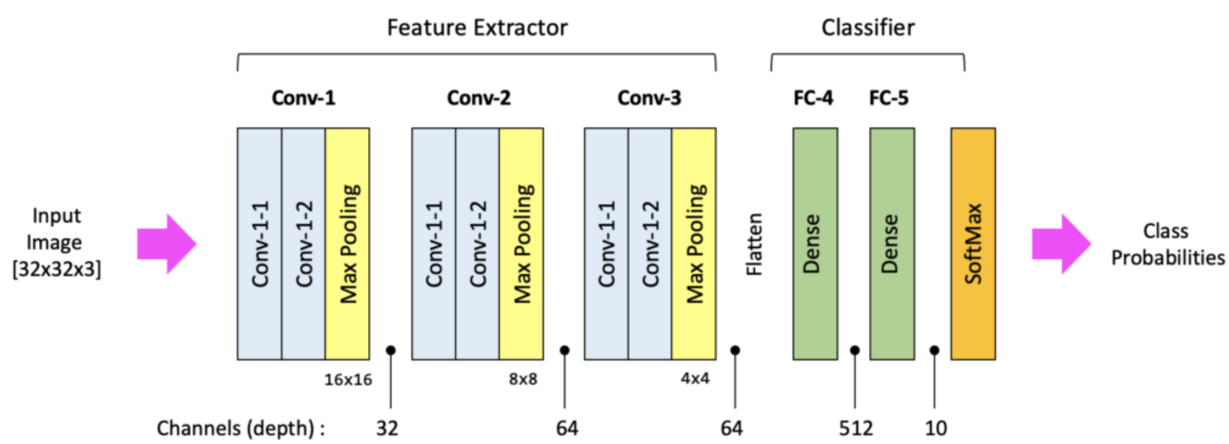

Figure 3. Example of a Convolutional Neural Network

Model Evaluation

Image classification Machine learning models are evaluated based on two key metrics: accuracy and loss. Accuracy is the percentage of inputs that the model correctly predicts. For example, if the model has 10 input images, and it correctly predicts 9 out of the 10 images, then its classification accuracy is 90%. As an absolute baseline, machine learning models should perform better than randomly guessing on data. For a binary image classification problem, the model is expected to have greater than 50% accuracy after training. This indicates that the model is learning patterns that contribute to an understanding of the problem past random guessing. A lower classification accuracy is expected the harder a problem is to learn. While a "good" classification accuracy varies greatly on the problem, a 90% accuracy is generally considered good. The second metric is loss. Loss is the numerical representation of accurate the model's prediction was. The better the prediction, the lower the loss. Thus, if a model is performing well, the loss trends towards zero while if the model is performing poorly, the loss trends positive. As the model continues to train, the loss should exponentially approach zero. Using these two metrics, a developer can plot learning curves that graph the loss and classification accuracy for a model. The model should be developed so that the classification accuracy increases towards 100% and the loss decreases towards zero.

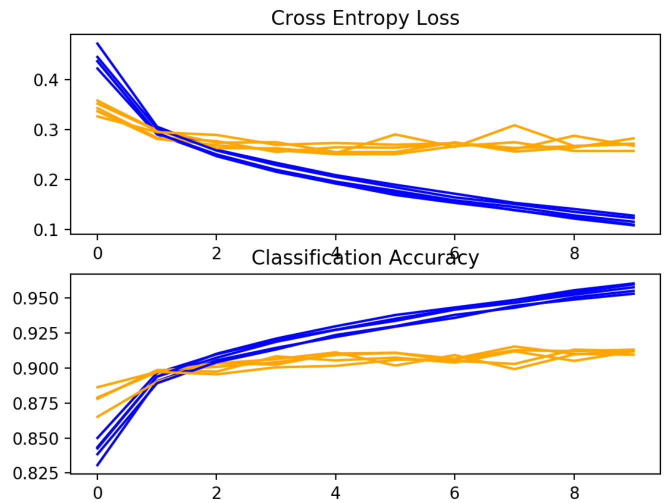

Figure 4. Example Training Curves for Loss and Classification Accuracy

Blue is the Training Data, and Orange is the Validation Data

CCO Image Classification Model

Resources Used

Machine learning models are very complex systems with intricate mathematical equations and matrix operations. Fortunately, there are many open-source libraries that handle these technical aspects and allow the user to focus on high level descriptions of the model. Python's simple syntax and large library support made it the clear choice for this project. Tensorflow is Google's open-source machine learning library which contains many tools to guide development of machine learning systems. In fact, Tensorflow is an end-to-end platform which supports data preparation, model training, and model deployment; additionally, Tensorflow supports most machine learning architectures including convolutional neural networks. The Keras application programming interface (API) accesses these library tools using easy to understand functions and processes. The Keras API is commonly used for simple machine learning models as it allows for quick and efficient implementation but disallows hyper-specific parameter tuning. In short, Tensorflow has numerous tools and functions to build machine learning models, and Keras is a high-level API that simplifies these tools into easy-to-use functions and structures. The CCO Image Classification model was implemented using Keras since the scope of the project was simple and did not require extensive tuning and functionality. Another library used was numpy, an array library for Python. Numpy is an array management software which allows for efficient memory usage and matrix operations in Python. Because images are represented using 2D matrices, numpy is an essential tool for implementing computer vision algorithms.

Dataset Structure

The dataset of a machine learning implementation is arguably the most important aspect of the project. Without a robust dataset, the model will not be able to recognize patterns, or worse, it will recognize false patterns from noise in the dataset. Additionally, without a sufficient dataset size, the model will not have enough examples to learn and extrapolate patterns. This project used 288 images to train and validate the model. Eight different CCO drums were available to image and record data with. 24 images were taken of each drum while empty, and 12 images were taken of each drum with various objects in them. The foreign items included a screwdriver, a bolt, a roll of tape, a flash drive, an electronic dongle, an SD card, a pen, a pipe fitting, and a USB head. These items were chosen to represent possible real-world items discovered in a CCO drum from workplace accidents or security risks; furthermore, items of various shapes and sizes were used. When developing a machine learning algorithm, models are usually trained on 80% of the data and validated on 20%. This allows the developer to evaluate the model's performance after training to quantify the performance of the model. This project's model was trained on seven drums and validated on the eighth which resulted in an 87.5/12.5% split. A larger training split was used because of the small dataset size to allow for the model to train on more data; moreover, one specific drum was held out to evaluate the model's performance on previously unseen data to more closely resemble real-world scenarios.

Model Structure

Convolutional Neural Network architectures are described by the layers used to define the model; the type, number, and order of layers used defines the model. The model architecture is found through repeated testing and evaluation on the dataset. As the model trains and is tested, graphs of the accuracy and loss are plotted. To develop the model, multiple architectures are developed, and a new graph is made for each architecture. The graphs are then compared to see which model architecture performs the best. This project chose a model with three convolutional layers each followed by a max pooling layer to learn the features of the model; three dense layers then classify the features and output a value between zero and one where zero corresponds to an empty drum and one corresponds to a full drum.

The input layer receives an RGB input image of 256x256 pixels. The convolutional layer filter sizes increase from 1x1 to 3x3, and 5x5 for all three layers. This allows the model to extract smaller defining edges early in the model and recognize larger distinct features later in model. Each convolutional layer is followed by a max pooling layer which is standard for CNN models. The first two dense layers have 256 and 64 neurons respectively. The final output layer is a single neuron which outputs a value between zero and one; this layer is the final classifier of the model with a threshold of 0.5. each layer uses the "relu" activation function except the final layer which uses a sigmoid function. The relu activation function is the best non-linear activation function for most machine learning models; however, they are only used for input and hidden layers. The sigmoid function is the most common output function for binary classification and is used for nearly every binary image classification problem. Finally, there are two dropout layers in the model to prevent overfitting. Dropout layers are simply layers that randomly deactivate a percentage of nodes on each training pass through. Dropout layers prevent the model from "memorizing" them images by removing the probability of the model learning statistical noise in the dataset and forcing it to learn general patterns. There is one dropout layer after the first pooling layer which randomly drops 20% of the inputs; another dropout layer is after the first dense layer which randomly removes 40% of the input neurons. The position and value of the dropout layers were chosen out of 13 tested models to ensure the best possible combination. In summary, the model is comprised of three convolutional and max pooling layers to extract features followed by three dense layers to classify images; two dropout layers are also used to prevent model overfitting.

Layers	Output Shape
2D Convolutional (32 filters)	(256,256,32)
2D MaxPooling	(128,128,32)
Dropout	(128,128,32)
2D Convolutional (64 filters)	(128,128,64)
2D MaxPooling	(63,63,64)
2D Convolutional (128 filters)	(59,59,128)
2D MaxPooling	(29,29,128)
Flatten	(107648)
Dense	(256)
Dropout	(256)
Dense	(64)
Dense	(1)

Table 1. Model Architecture

Design Process

Previous models were developed and tested. The first model was a single convolutional and max pooling layer followed by two dense layers. This model was able to general certain features of the problem, but overall, it performed poorly with only about a 70% accuracy. The model was then tested with increasing convolutional and dense layers. A max pooling layer placed after every convolutional layer performed the best in testing, so this design choice was kept in the final model. The current three convolutional/three dense layer architecture was eventually chosen as it provided enough layers for the model to learn much of the problem but did not waste memory with extraneous layers. Additionally, with this architecture, the mo del began to noticeable overfit on the data; therefore, dropout layers were added to force generalization of the problem. In all, the final model was developed after testing several different architectures and tuning the parameters to best fit the data.

Results

Figure 5. Model Successfully Classifies CCO Drum

The results of this model are promising but still limited by the small dataset size. A common "small" dataset for neural networks is in the thousands of images; however, the current dataset is limited to 288 pictures. In future work, a larger dataset is necessary to ensure confidence in the final model. For the current development, results are noticeably varied as the graph lines jump significantly; even one wrong prediction drops the accuracy by 2.78%. Nevertheless, the final model had an average of 85% classification accuracy. The final loss was more varied but stayed around 0.4-0.8. The loss metric is concerning, as 0.61 is the threshold for randomly guessing in this binary classification problem. Loss above 0.61 indicates that the model is not accurately learning the problem, and the variability of the loss value suggests that the model might be memorizing the problem more than generalizing. Moving forward, a larger dataset size will reveal if the loss value varies due to a lack of data values or if it indicates a bigger problem with the model. However, the validation accuracy is encouraging and suggests that the model can somewhat accurately classify drums as empty or full. In all, the model is a great baseline for future research. The foundation for future research exists, and continued testing and development will explore and demonstrate the capabilities of machine learning image classification for implementation in automated systems.

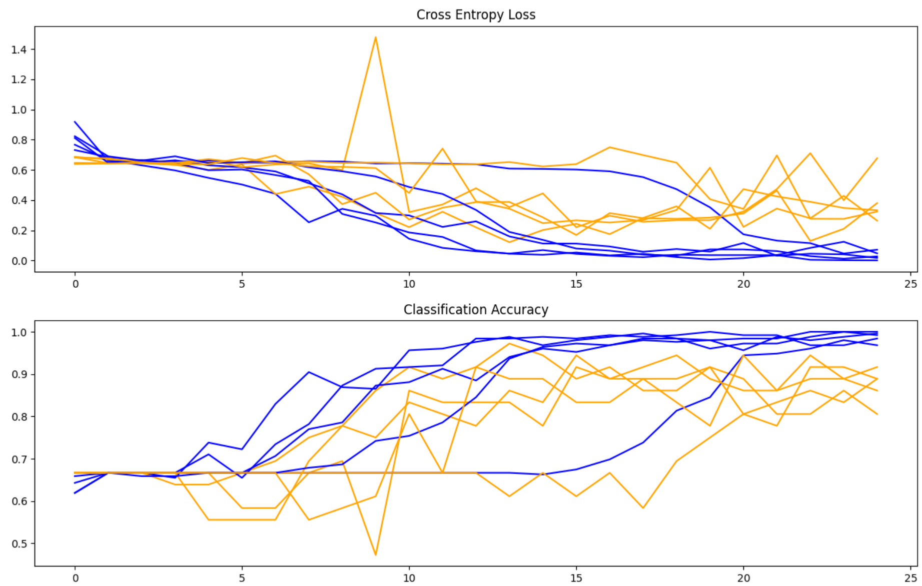

Figure 6. Training Curve for Image Classification Model