Using Machine Learning to Analyze Image Data from Advanced Manufacturing Processes

Shubham Patel, James Mekavibul, Jami Park, Anchit Kolla, Ryan French, Zachary Kersey, Gregory C. Lewin

Motivation

Additive Manufacturing

Additive manufacturing (AM), commonly known as 3D printing, has competitive advantages over other processes in reducing product manufacturing times and costs, but does not reliably print predictable and repeatable builds. Laser powder bed fusion (L-PBF) printers build parts by repeatedly melting layers of metal powder with a fine, powerful laser.

Dross

Dross is a defect that occurs in metal prints, typically at overhangs. It forms when excess metal powder gets melted onto the print. These impurities can be seen as rough edges or excess material on the outsides of the print. Dross formation increases the variability in prints, deterring industry-wide adoption of metal 3D printing.

Impact

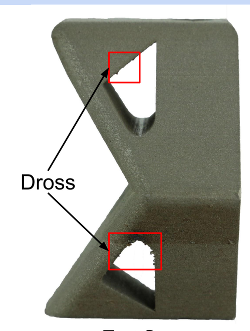
Currently, the only method of identifying dross at CCAM, the Commonwealth Center for Advanced Manufacturing, is through human detection. Algorithms and models can circumvent human error and increase detection efficiency. Developing a tool that may one day be used in real time during prints can save time, money, labor, and materials.

How Can Dross be Automatically Identified in L-PBF Prints?

Automatically identifying dross with a computational model would ultimately be used to identify dross in real-time. As the 3D printer is building the part, the layer image that is taken could be transformed and input into the model, which would identify any dross. Though real-time implementation is the end goal, this research team focused on creating the computational model that would process the images. Below is a diagram and explanation of how the team implemented a machine learning model that automatically identifies dross in L-PBF prints.

1	2	3	4
Acquire Data	**Transform Data**	**Label Data**	**Train and Test Model**
Print test parts and acquire layer images	Change the perspective of the layer images and build 3D models	Identify and label dross on images using 3D model and physical part	Train a model to identify dross in images and test its efficacy

1. Acquire Data



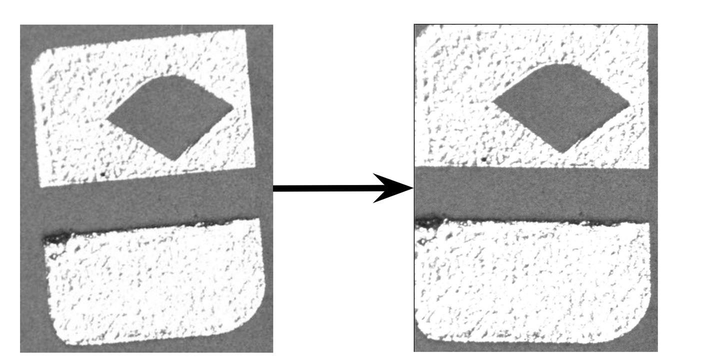
Test Part

Digital model of test part using 300 images

The stainless steel test part, printed with CCAM's EOS M290 L-PBF printer, is 1.5"x2"x2.25" and designed with sharp angles and steep overhangs in order to generate print data that included dross.

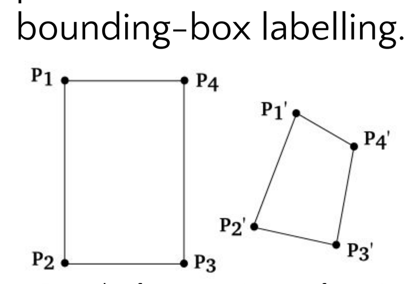
During the printing process, sensors collected numerous forms of data, including IR, image, and positional data. We chose to focus on image data, since dross is easy to recognize visually during the labelling process. A camera inside the printer took an image of the build plate after each new layer is deposited. This process generated 3179 images in total. A subset of the images, around 300 in total, was used. These images were selected as they were associated with portions of the print that were known to contain dross.

2. Transform Data



The image data provided by CCAM was taken from a non-normal point of view above the print. In order to adjust the images to a top-down view, homography transformations were implemented.

This homography was implemented in Python via a perspective transformation using OpenCV. With this transformation, all the images will be standardized from a single point of view, and allows for much easier bounding-box labelling.



Example of a perspective transformation

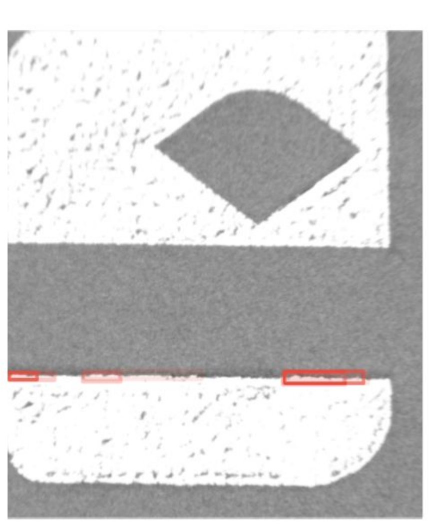
OpenCV

3. Label Data

The team manually labelled 301 images, three times each. LabelBox, a third-party tool, allowed the team members to quickly label pictures as "Poor Picture", "No Dross", or draw bounding boxes around dross directly on the image. The intersection of all bounding boxes was used to train the neural network.

	Dross	No dross/Poor picture	Total2
Training	80	116	196
Test	41	64	105
Total	121	181	301

Of the 301 images, 121 contained dross. Two-thirds of the images were used for the training set, one-third for the testing set.



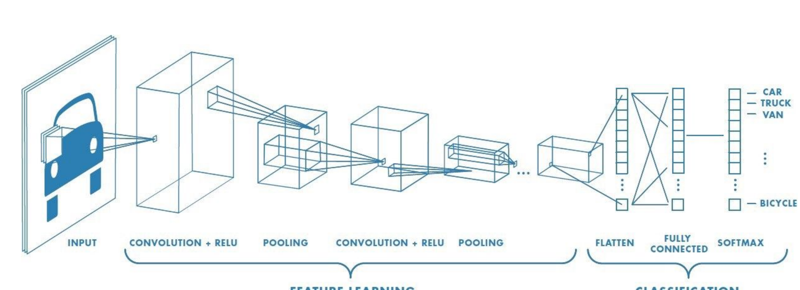
Labelled image with intersection of three iterations of bounding boxes. The red overlaps indicate intersections.

By comparing the physical part & digitally generated models with the images, we highlighted certain drossy areas. Specifically, the team looked for dark areas along edges and identified dross as extruding white bumps from the "normal" surface of the part.

We then split this data into 2 portions: two-thirds of the data was for training (used for improving the network), and one-third was for testing (used for evaluating the model).

4. Train and Test Model

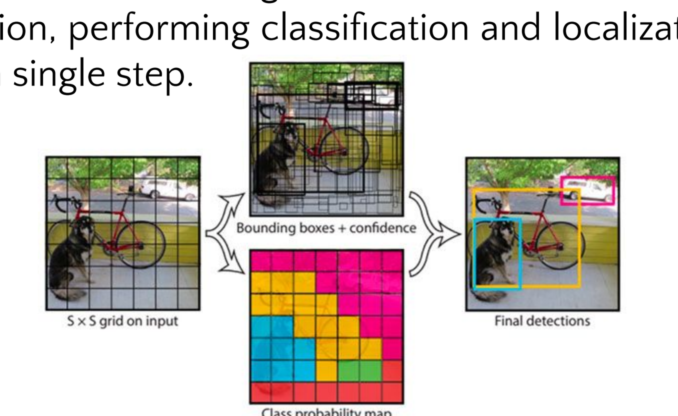
The labeled data is fed through a convolutional neural network (CNN), which utilizes classification data in order to identify different parts of an image. The CNN is used to create a filter, which is then used to classify unlabeled images during the testing phase. This type of model is commonly used in computer vision.



Example of CNN model structure. Breaks down an image into convolutions.

A CNN consists of multiple convolutional layers, which allow for each entry to analyze a specific subsection of the image.

The YOLO v3 (You Only Look Once) object detection system allows the network to save time and processing power by focusing only on search specific areas for dross instead of running a general classification of the entire image. Also, YOLO's fast classification times mean it is ideal for real time object detection, making it potentially useful in real-time prints. Rather than repurposing classifiers/localizers, YOLO divides the given image into regions and generates bounding boxes for each individual region, performing classification and localization in a single step.



Example of implementing YOLO on a picture with three different classes

Results

The YOLO network was trained for 2000 iterations on the dross and nondross images. In each iteration, the network changes its weights to improve the output to better approximate the input data. The network was trained with a batch size of 64, momentum of 0.9, and decay of 0.0005. Batch size refers to the number of samples that would be propagated through the network through each iteration, and momentum prevents the network from getting stuck in a local minima. Decay is the value the determines how much outlier weights in the network are penalized to 0, allowing for better generalization.

With a confidence threshold of 0.50, the results from our training achieved an average precision (mAP) of 67.69% and a recall of 83%. Precision tells how many of the identified images actually contained dross. Recall tells how many of the images that actually have dross were correctly identified.

68% Precision

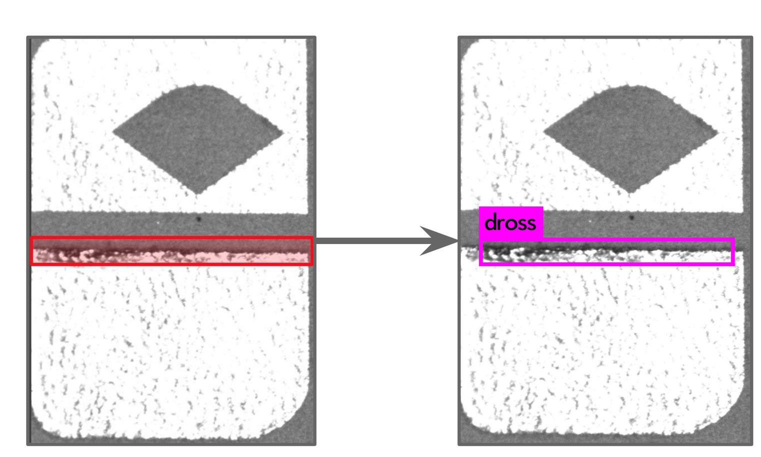
83% Recall

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

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

TP = True positive
TN = True negative
FP = False positive
FN = False negative

Formulas for Precision and Recall



Labelled image in training set

Output label from CNN

Conclusion

The 68% precision and 83% recall rates suggest that the model can identify dross with some certainty. The average time for the network to detect dross in an image is 0.088 seconds, meaning that it is also possible to utilize this network for real-time monitoring of prints for defects, since many printers take a longer time to build each layer. For real-time implementation, the client will take transformed layer images of an AM print and pass them into the model. The output of the model will be the same images passed in, but with dross labeled by bounding boxes.

Ultimately, the success of the model indicates that an automatic method of identifying dross in L-PBF prints is feasible. The results suggest that, with future work in improving the model, the model can be implemented into a larger system that identifies dross in real-time.

Future Work

On an implementation-specific level, the project can be improved by labeling more data that can train the model. Furthermore, other neural networks could be explored, such as SSD (Single-Shot Detection), to see if they yield better results. The project is limited by only being tested with a single part's layer images. Future iterations of this project should be tested across multiple prints to ensure extendability. Ultimately, this project will be integrated into a real-time system, which requires much more future work.

Acknowledgements

This project was supported by the Commonwealth Center for Advanced Manufacturing (CCAM). A special thanks to Kyle Snyder and Arthur French for assisting with the design, creation, and experimentation of the image recognition model, without which the project would not have been possible. A special thanks to Keith O'Connor, Joseph Fitzsimmons, and Justin McAuliff for assisting with the SIEDS paper and presentation. A special thanks to LabelBox for giving this capstone team an educational license to their product.