Product defects detection using Deep Learning

Phuong Anh Nguyen
Columbia University
pn2394@columbia.edu

Abstract

Defects can result in significant waste and expenses to manufacturing businesses, and detecting surface defects is a challenging task that has received considerable attention in recent decades. Traditional image processing techniques can address certain types of problems, but they struggle with complex backgrounds, noise, and lighting differences. Deep learning has recently emerged as a solution to these challenges, driven by advances in computing power and the availability of large datasets. This research paper proposes the use of deep learning models to detect surface flaws, with potential benefits for manufacturing businesses' quality assurance.

Dataset Description

Northeastern University (NEU) surface defect database (*)

Author: Kechen Song and Yunhui Yan [1], [4], [5]
Link to download: Google Drive / Google Drive (Backup link)

[1]The dataset has 1,800 grayscale images with annotations: 300 samples each of six different kinds of typical surface defects of the hot-rolled steel strip: rolled-in scale (RS), patches (Pa), crazing (Cr), pitted surface (PS), inclusion (In) and scratches (Sc). The original resolution of each image is 200×200 pixels.

Figure 1 shows some sample images of 6 kinds of typical surface defects:

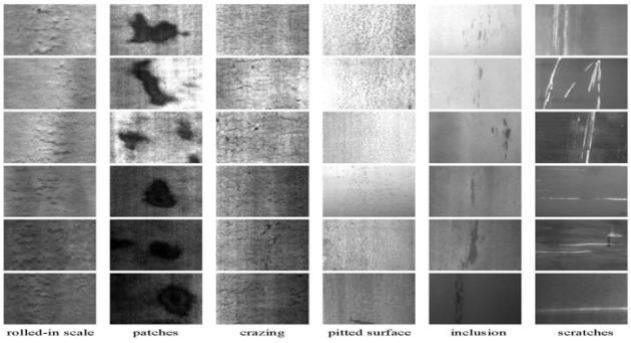


Figure 1

In short, the NEU surface defect database includes two difficult challenges, i.e., the intra-class defects existing large differences in appearance while the inter-class defects have similar aspects, the defect images suffer from the influence of illumination and material changes. The dataset includes annotations which indicate the class and location of a defect in each image. For each defect, the yellow box is the bounding box indicating its location and the green label is the class score.

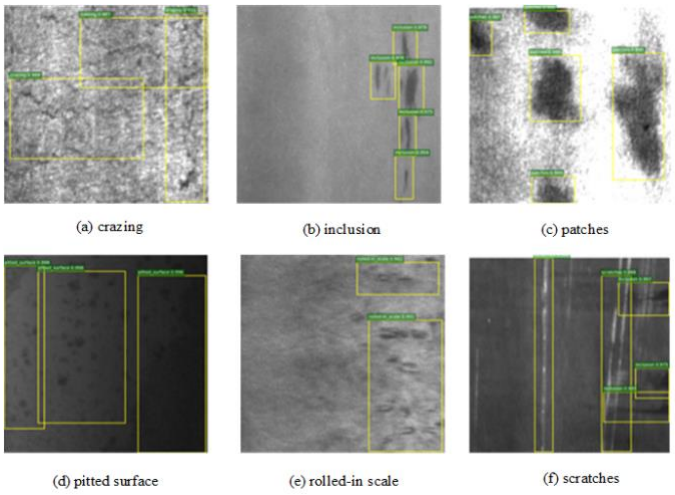


Figure 2. Examples of detection results

(*) There's also a Kaggle challenge about this dataset but only 4 people have attempted it and haven't finished.

Methodology

I chose Convolutional Neural Network (CNN) since it's a specialized model for image and video processing. It has also been demonstrated to be an effective tool for automated identification of defects (Rahman, 2022) [3].

I also proposed a model that integrated Recurrent Neural Network (RNN) and Convolution Neural Network (CNN). Recurrent neural network (RNN) is a special type of neural network that is well-suited for processing sequential data. This makes them ideal for defect detection, as defects are usually observed in pixel sequences. RNNs can also assist in accelerating flaws detection by "remembering" information in previous frames and use them to predict objects in future frames.

Inspired by Liang and Hu (2015)[2], I built a Recurrent CNN (RCNN) for defects identification by incorporating recurrent connections into each convolutional layer, with the output of each CNN serving as the input to the RNN.

With these extra connections, the network can evolve over time, with neighbor units influencing each other. This property contains context information for an image, which is essential for object detection.
However, this model demands a lot of processing power and could not be implemented on my laptop with low GPU capacity.

Experiment

Environment: I ran my models in the Keras R interface, which uses Tensorflow as its default back-end.

Baseline model: I first fitted a simple 2D Convolutional Neural Network with 3 hidden layers, 32 hidden units each, dropout = 0.5 in each layer. I took 20% of the training data to be the validation set.

The Loss function for this model is **cross-entropy**, and the optimizer is **RMSProp**, which are similar to the other models I used later.

The model did not perform well. It achieved an average Training accuracy of 18.25% and a Validation accuracy of 13.70%, which are not significantly different and indicate that the model did not overfit on the training data. The Validation losses are approximately 0.001 more than the Training losses.

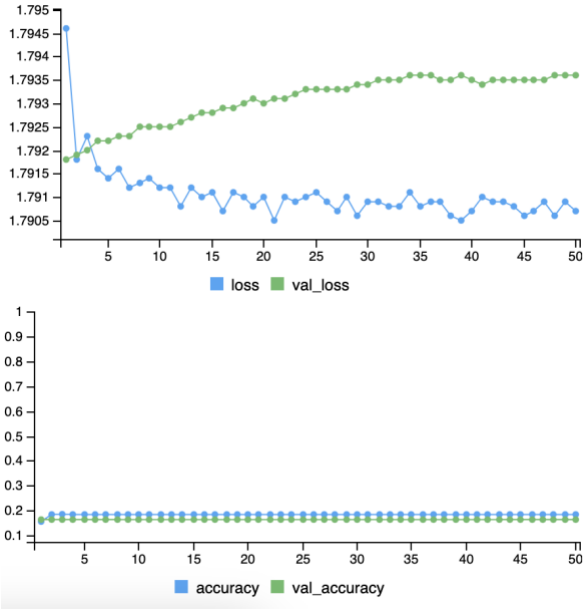


Figure 3. Training & Validation Loss & Accuracy of Baseline model

Deep CNN: I increased the model complexity by adding 2 more layers (5 layers in total), increase the hidden units in each layer to 218, dropout = 0.5 in each layer to reduce overfitting. I also increased the kernel size from (3 x 3) to (5 x 5) to enable the model to learn more complex features.

The model achieved much better results than the baseline model, with decreased Validation Loss and increased Validation Accuracy after about 25 epochs.

Due to the limitation in computational power, I could not add more layers or hidden units.

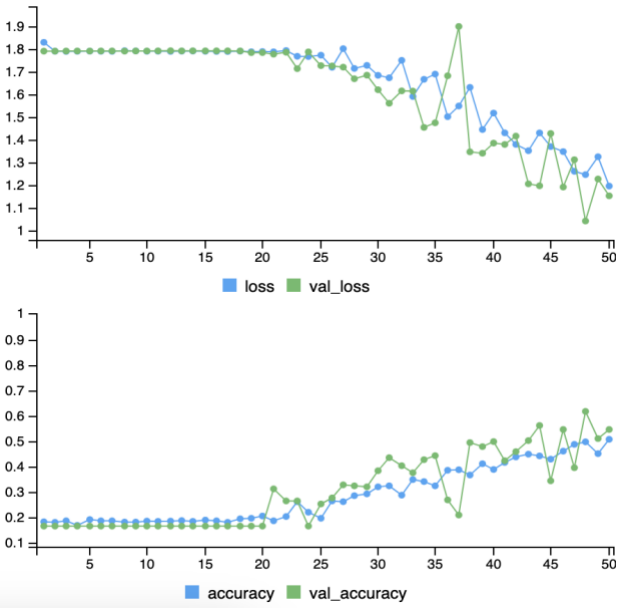
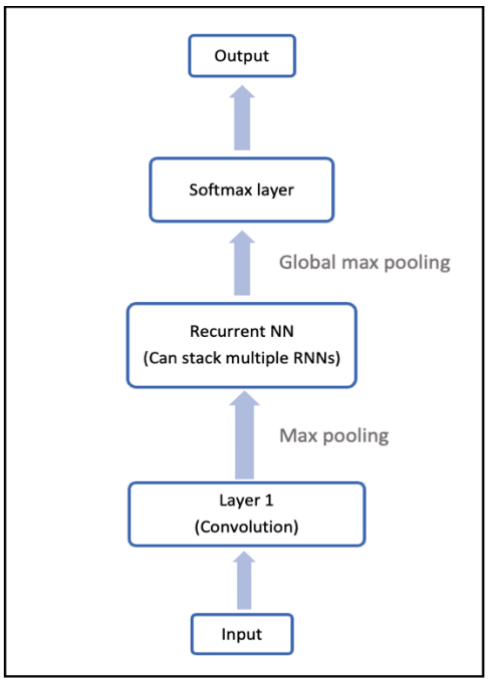


Figure 4. Training & Validation Loss & Accuracy of Deep CNN model

Combine RNN with CNN (RCNN):

I constructed a RCNN architecture as followed:



The network consists of first a convolutional layer to save computations, which is followed by a max pooling layer. On top of that one or more RNNs can be used ("stacked RNNs"). Stacked RNNs generates more representational power than a single-layer RNN. Finally, a global max pooling layer and a softmax layer are used.

Results

Model	Testing Loss	Testing Accuracy
Baseline model	1.7965	13.70%
Deep CNN	1.2621	47.40%
Recurrent CNN Network (RCNN)	Not enough computational power to run	

The 5-layer CNN model has a lower Loss and a higher Testing Accuracy than the baseline CNN model, which is a substantial improvement. However, the total accuracy is still low. One of the reasons can be the model is not deep enough. However, due to computing power limitations, I was unable to construct a deeper neural network to support this theory.

Conclusion

The deeper neural network was able to capture more data and has a greater accuracy than the baseline simple CNN. If the Recurrent Convolutional Neural Network can be implemented, I believe it will also yield a better result.

References

[1] K. Song and Y. Yan. (Nov 2013). A noise robust method based on completed local binary patterns for hot-rolled steel strip surface defects. *Applied Surface Science, vol. 285, pp. 858-864*. (paper)

[2] Liang, M., & Hu, X. (2015). Recurrent convolutional neural network for object recognition. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 3367–3375. https://doi.org/10.1109/CVPR.2015.7298958

[3] Rahman, M. F. (2022). A Convolutional Neural Network (CNN) for Defect Detection of Additively Manufactured Parts. *Volume 2A: Advanced Manufacturing*, V02AT02A010. https://doi.org/10.1115/IMECE2021-70500

[4] Yanqi Bao, Kechen Song, Jie Liu, Yanyan Wang, Yunhui Yan, Han Yu, Xingjie Li. (2021). Triplet- Graph Reasoning Network for Few-shot Metal Generic Surface Defect Segmentation. *IEEE Transactions on Instrumentation and Measuremente*. (paper)

[5] Yu He, Kechen Song, Qinggang Meng, Yunhui Yan. (2020). An End-to-end Steel Surface Defect Detection Approach via Fusing Multiple Hierarchical Features. *IEEE Transactions on Instrumentation and Measuremente, 69(4),1493-1504*. (paper)

Capstone Project (Spring 2023)

• Construct a consistent, accurate pricing model that accounts for numerous variables (e.g., ceramic costs, manganese prices), forecast production and energy costs during actual production

• Develop a user interface for non-Python users to enter key variables (e.g., customer code, item number, expected energy index trend) and generate an appropriate customer-specific pricing list