CNNs for Bulk Material Defect Detection

Akshay Aravindan
akshay14@stanford.edu

Harrison Greenwood
hgreenw2@stanford.edu

Aakriti Varshney
aakritiv@stanford.edu

Department of Mechanical Engineering
Stanford University

Abstract

Detecting defects in bulk materials is one of the challenging problems for industries worldwide and is currently very manual and time consuming. This project is aimed at developing a model that can perform surface defect detection in bulk materials, particularly steel and plastic. Through this study, defects in steel were detected by applying transfer learning to 12568 steel images using a pretrained ResNet18 and ResNet50 Convolutional Neural Network (CNN) and it was hypothesized that these networks would perform well on a small dataset of similar plastic images. After making suitable changes to the models, a test accuracy of 87% was obtained with the ResNet18 model while an accuracy of 90.2% was obtained using the ResNet50 model on the steel dataset. However, while applying both these models on the plastic dataset, we only obtained a 60% accuracy, indicating that our initial hypothesis was not accurate.

1 Introduction

We propose to investigate the problem of predicting defects in steel. As a team of mechanical engineering students, we're familiar with the importance of quality assurance for bulk materials; shipping steel can be extremely expensive (both in time and cost), and if the steel performs differently than expected, the purchaser takes on large risks; steel that underperforms can damage existing parts in the project and/or equipment used to work it. Manufacturers must also meet federal regulations when supplying steel. Steel suppliers are therefore under high pressure to ensure their quality assurance is both robust and inexpensive.

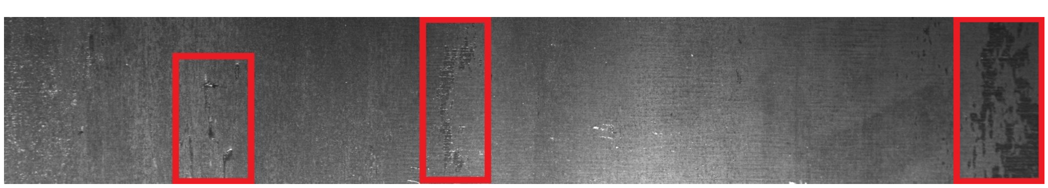


Figure 1: Sample image with defects in Steel

From Figure 3, we see a sample image from the training dataset that has defects. (As isolated by the red box) while figure 1.2 shows a defect free steel plate.

CS230: Deep Learning, Fall 2019, Stanford University, CA. (LateX template borrowed from NIPS 2017.)

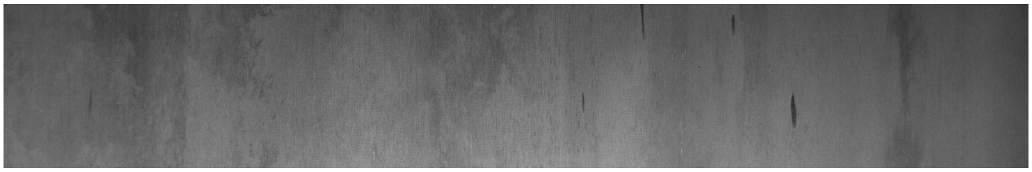


Figure 2: Sample image with no defects in Steel

Detecting defects in steel plates can be expensive, time consuming, and prone to error, especially with the scale of a steel mill. Generally, human operators must manually inspect the product, possibly with the aid of specialized equipment. A mechanized approach could be more efficient. Currently in large scale manufacturing, it is increasingly expensive to ensure this level of quality assurance manually. By automating this process, we foresee potential savings for all such manufacturing industries.

However, this issue is not isolated to steel plates alone. There are several such engineering materials that can benefit from defect detection. We would first use transfer learning on a pre-existing image classification network such as ResNet18 and ResNet50, to create a binary classification model that is able to detect steel defects. This model can, in turn, be applied to other engineering materials like aluminum, brass, bronze etc. to be able to detect defects with no change in network architecture. Thus, in our project we hypothesize that by using a ResNet model that is pre-trained with high accuracies on steel defects we will be able to use this model as is, on other bulk materials like plastic and detect defects with similar accuracies.

2 Related work

Steel defect detection alone has been done under a variety of contexts. Of course, the traditional method was manual, human inspection, but in recent years deep learning algorithms have proven successful. One paper described steel rail defect detection on a train. Another detected casting defects using X-ray imaging. Since these papers were written with very specific tasks in mind, they weren't applied to an array of materials. And while work has been done to create generalized defect detectors capable of finding inconsistencies in images with varying backgrounds, there hasn't been significant success in creating a network capable of distinguishing bulk material defects across multiple real engineering materials.

3 Dataset and Features

The dataset provided in the kaggle competition (`https://www.kaggle.com/c/severstal-steel-defect-detection/data`) contained about 12,568 grayscale images of steel surfaces with and without defects. This dataset was originally intended for segmentation, but for the sake of scope for this project, we converted the encoded pixel annotations into binary classification (1 for defect, 0 for defect-free). Training was done on this steel image dataset, so it was split 80-10-10 into train, val, and test. The second engineering material considered was plastic. A set of ~500 grayscale microscope images of defective plastic were obtained from kolektor(`https://www.vicos.si/Downloads/KolektorSDD`).

The steel dataset images were resized from 1600 by 256 pixels to 512 by 512 with padded black pixels. Similarly, plastic images were resized from 500 by 1266 to 512 by 512 for compatibility with the model used. To human eyes, the resulting resolution was more than adequate for identifying defects, however, there were confusing aspects to the steel dataset. What might be called a defect by one viewer wasn't necessarily considered a defect by Severstal.

4 Methods

To build a model capable of distinguishing generic defects in a variety of materials with a relatively small training set, we opted to use transfer learning of an existing model. Models considered were pretrained ResNet18, ResNet50, and ResNet101 built into pytorch. Only the last layer of the ResNet was removed and replaced with a new final, fully-connected layer for binary classification. The

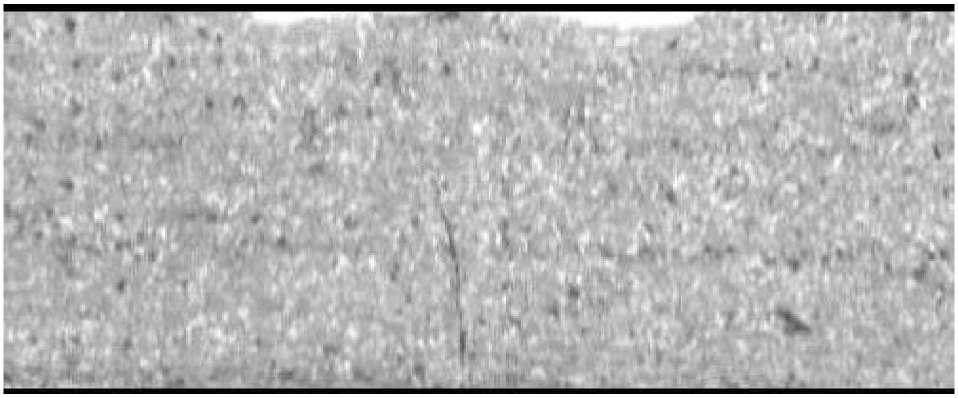


Figure 3: Plastic image with a defect lower-center

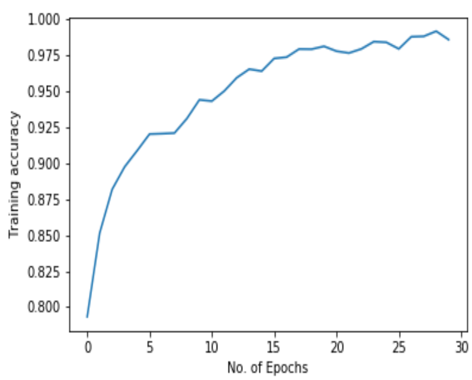
number of frozen layers was the only hyper-parameter of transfer learning that was tuned. Binary cross-entropy was used for the loss function.

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

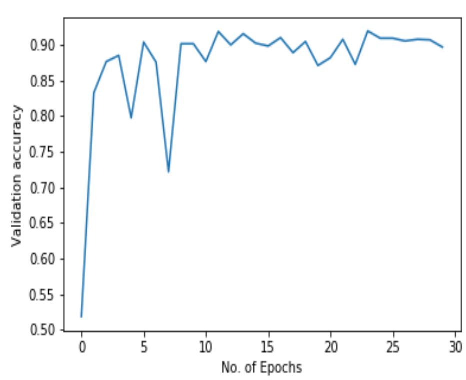
Adam was used for the optimizer. Accuracy was used as the metric of evaluation of the model. Additionally, learning rate and regularization parameter were also tuned to improve convergence and solving the variance problem respectively.

5 Discussion

Initially a ResNet18 model was used with only the last 5 layers unfrozen. Running this model to convergence yielded only around 70% training accuracy- insufficient for a binary classifier. To combat the issue of high bias, we unfroze more parameters. Training the last 16 layers of the ResNet-18 performed well and yielded an accuracy of around 97% on the train set, 90% on the val set, and 87% on the test set.



(a) Training accuracy



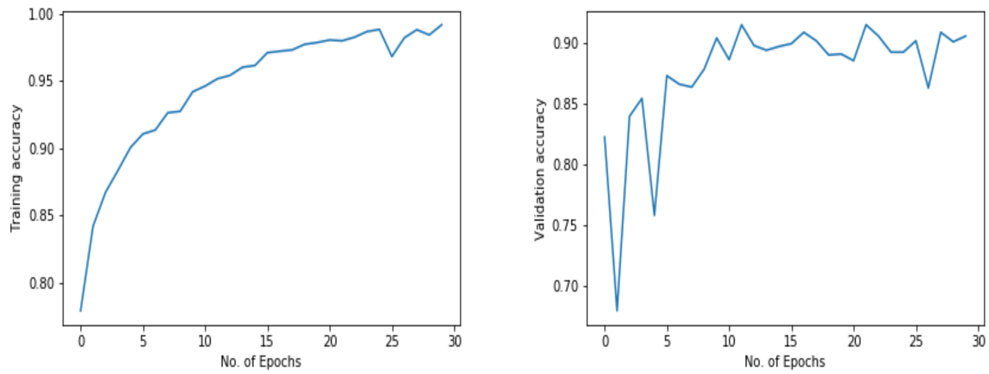
(b) Validation accuracy

Figure 4: Training and validation accuracies on ResNet18 with the last 5 layers unfrozen

The high-level feature-detectors produced by the deep layers of the pretrained model were not useful for identifying defects. Unfreezing more layers prior to training greatly reduced the bias issue.

To further improve our performance, we decided to experiment with a deeper ResNet. The last 50 layers of ResNet101 were unfrozen and trained. Once again, the bias problem led to a poor accuracy of only 74%. Unfreezing the last 98 layers of the model resulted in an accuracy of 94.7% on the train set and around 90% on the dev set. This brought up the issue of diminishing returns. Greatly increasing the depth of the network only marginally increased the accuracy of the model.

To strike a balance, a ResNet50 model was used with the last 48 layers unfrozen. This was a good compromise between computational cost and increased accuracy and a train accuracy of 99% , 92% on the dev set and 90.2% on the test set was obtained.

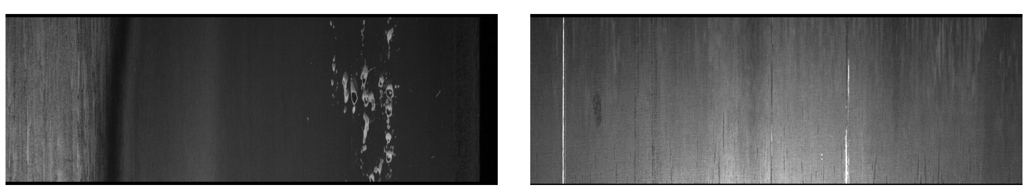


(a) Training accuracy

(b) Validation accuracy

Figure 5: Training and validation accuracies on ResNet50 with the last 48 layers unfrozen

One of the other important considerations here, is the hypothesis of a high bayes error on the steel defect dataset. It was difficult to predict if some of the images in the dataset actually had defects despite the experience of being mechanical engineers. This could potentially be a reason for the saturation of model accuracy. Additionally, since the initial dataset was for a segmentation problem given by Severstal, the defects were annotated better in the dataset using encoded pixels which could allow the model to understand better what defects looked like in the dataset.



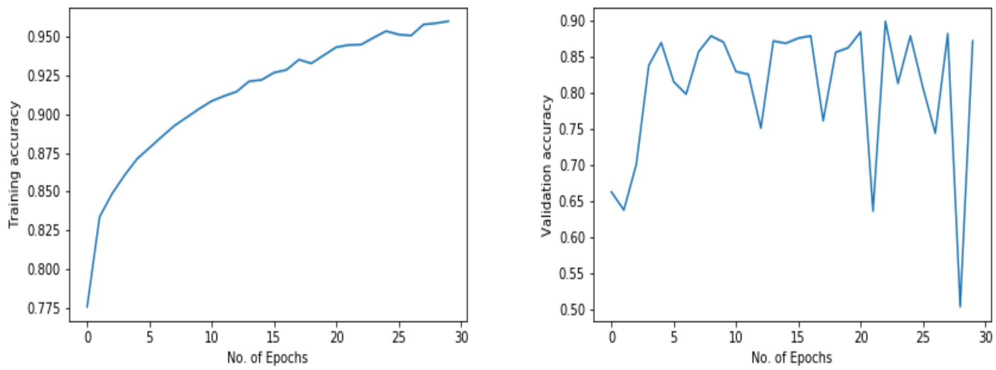
(a) Non-defective steel

(b) Defective steel

Figure 6: Examples of non-defective and defective steel images from the dataset demonstrating a high human error

One of the other issues with the model performance was the high variance problem causing validation accuracies to be lower than training accuracies. In order to address this, the regularization parameter was increased from 1e-5 to 1e-3 to improve regularization and hence reduce variance. But unfortunately, even after this hyperparameter tuning, the validation accuracy was still found to only be 86% when the training accuracy was 96%.

With a well trained, high accuracy-yielding ResNet on steel defects, the actual problem at hand could be tackled - evaluating this trained model's performance as a bulk material defect detector. As a starting point, the pretrained model was tested on the plastic dataset and an accuracy of 60.2% was obtained. The initial hypothesis that training a model on one material would result in high accuracy defect detection on all bulk materials was thus invalidated based on this model's performance. However, the reason for the low accuracy could be conceptual or attributed to this particular ResNet model. Further investigation would need to be done.



(a) Training accuracy

(b) Validation accuracy

Figure 7: Training and validation accuracies on ResNet50 with the last 48 layers unfrozen with a higher value (1e-3) of the regularization parameter

6 Conclusion and Future Work

In conclusion, transfer learning was performed using a pretrained ResNet CNN on a dataset of steel defect dataset. Hyperparameters were tuned to obtain high accuracies for binary classification of steel defects. This was then applied as is to plastic defects dataset to observe performance.

Model (on Steel)	Test Accuracy
ResNet18 (Last 5 layers unfrozen)	70.2%
ResNet18 (Last 16 layers unfrozen)	87.0%
ResNet101 (Last 50 layers unfrozen)	70.5%
ResNet101 (Last 98 layers unfrozen)	90.6%
ResNet50 (Last 48 layers unfrozen)	90.2%

The table above shows a summary of the results obtained on the steel dataset with different networks and hyperparameters.

Based on the results obtained, the first step would be to perform an error analysis on the classification and understand better where the misclassification is occurring and if there are common trends for the misclassification that validates our theory of the Bayes error being high.

Additionally, understanding why our model did not perform well on the plastic dataset would be of importance. It would be imperative to understand if this was a characteristic of our model or if conceptually our hypothesis has no basis. Extending our hypothesis to other bulk materials other than plastic would also help to analyze the model performance on overall bulk materials.

7 Contributions

All members of the team contributed equally. Harrison Greenwood found datasets, performed all data preprocessing, constructed the dataloader, and helped train the model. Akshay Aravindan and Aakriti Varshney focused on choosing the right model, modifying its layers for transfer learning and applying the model to the loaded data. They also worked on tuning some hyperparameters to improve model performance.

References

[1] Ferguson, Max, et al. "Automatic localization of casting defects with convolutional neural networks." 2017 *IEEE International Conference on Big Data (Big Data)*. IEEE, 2017.

[2] Wang, Tian, et al. "A fast and robust convolutional neural network-based defect detection model in product quality control." *The International Journal of Advanced Manufacturing Technology* 94.9-12 (2018): 3465-3471.

[3] Shang, Lidan, et al. "Detection of rail surface defects based on CNN image recognition and classification." *2018 20th International Conference on Advanced Communication Technology (ICACT)*. IEEE, 2018.

Code

The code used for this project can be found here: `https://github.com/hgreenw2/CS230`. It has been written using the Pytorch framework.