A Comparison of Convolutional Neural Networks, Random Forest and K-Means/TSNE for Classification of Defects in the NEU-CLS Dataset

Nolan McCarthy[1], Fatty Moo Moo[2]

1. Carnegie Mellon University, 2. Lab Assistant/Cat

1. Introduction: For image classification problems convolutional neural networks (CNNs) have been the go-to method for academia and industry. CNNs have shown high levels of accuracy on very difficult image classification problems such as the ImageNet challenge. CNNs have made inroads from computer science to materials science in recent years and have shown great promise.

The North Eastern University Surface Defects Database (NEU-CLS) is a database containing images of six different classes of steel defects with 300 images of each defect class, a sample of which is shown in Figure 1. The NEU dataset is the equivalent of the Iris dataset for machine learning in materials science, with multitudes of papers being published on its analysis. Many of the efforts in classifying the images has been done using CNNs, with CNN transfer learning showing accuracy of > 98%[1][2].

In this paper I will present a comparison of a CNN of my own design with two novel algorithms which use predetermined image filters, one supervised using Random Forest (RF), and another unsupervised using TSNE and K-Means.

In this paper I will show that methods other than CNNs can achieve similar accuracy with reduced complexity, computer hardware requirements, and without the “black box” nature of CNNs.

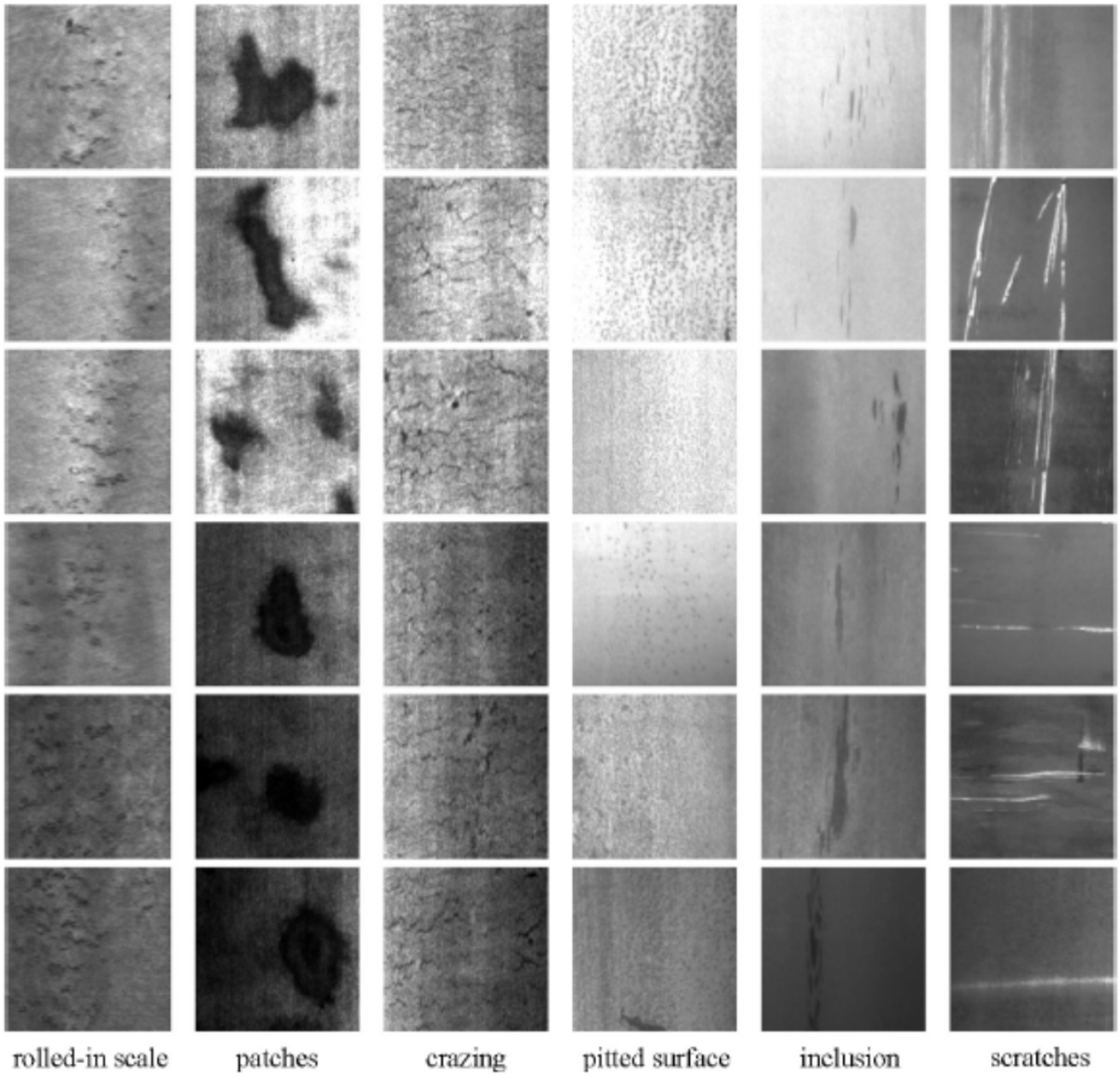

Figure 1. Samples of each defect class in the NEU dataset. Source: [1]

2. Methods:

2.1 Modified CNN

CNNs have the benefit of being translationally invariant, however they are not rotationally invariant. In Figure. 1 we can see in the “scratches” column that textures in the steel pattern can have an arbitrary rotation, this calls for a small modification to the image preprocessing. Some work has been done in pure computer science to make rotationally invariant neural networks[3] which have shown success in identifying

arbitrarily rotated images. Our method involves transforming the image from Cartesian space into log polar space in other words the map (x,y) -> (arctan2(x,y),log(sqrt(x^2 + y^2)) where x and y are the pixel positions relative to the center of the image. In experiments, this resulted in a significant performance boost compared to a CNN without the log-polar transform. An example of a log polar transformed image is shown in Figure. 2.

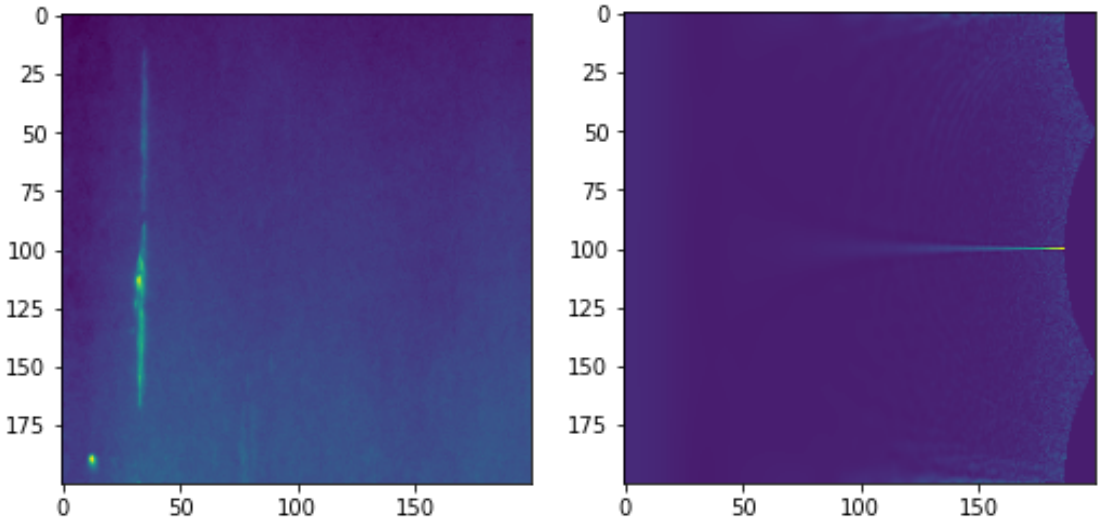

Figure. 2: (left) an image of a scratch. (right) the same image transformed into log-polar coordinates. Any rotation in cartesian coordinates will become a translation in log polar coordinates.

The network itself was composed of 3 hidden convolutional layers, the first convolutional layer contained a 20x20x200 kernel, the second contained a 1x1x100 kernel, and the third had 1x1x6 kernel, each with ReLU activation and a 20% dropout rate, followed by a average pool and softmax across all six outputs.

2.2 Feature Vectors Made from Predetermined Image Filters

Compared to the CNN which learns the kernels which extract features, this method applied image kernels that were guessed beforehand to extract important image information.

The Sobel operator has been a staple of machine vision for decades, the Sobel operator is nothing but a discrete first derivative in either the x or y direction, the kernel of the sobel operator is shown in Figure. 3.

$$\mathbf{G}_x = \begin{bmatrix} +1 & 0 & -1 \\ +2 & 0 & -2 \\ +1 & 0 & -1 \end{bmatrix} * \mathbf{A} \quad \text{and} \quad \mathbf{G}_y = \begin{bmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} * \mathbf{A}$$

Figure. 3: The Sobel Operator is the equivalent of a discrete derivative of an image, it is commonly used for edge detection. Gx is the gradient in the x direction Gy is the gradient in the y direction.

The x and y gradient can be turned into the total gradient by the equation sqrt(Gx^2 + Gy^2) additionally the direction of the gradient can be found by arctan2(Gx,Gy). To turn each image into a feature vector we took the mean and standard deviation of the untransformed image, the mean and standard deviation of the x gradient, y gradient, total gradient, gradient direction, and the second derivative of the image which was obtained by applying the x and y sobel operators twice and taking the root sum square, producing a feature vector containing 14 columns. An example of the filtered images is shown in Figure. 4.

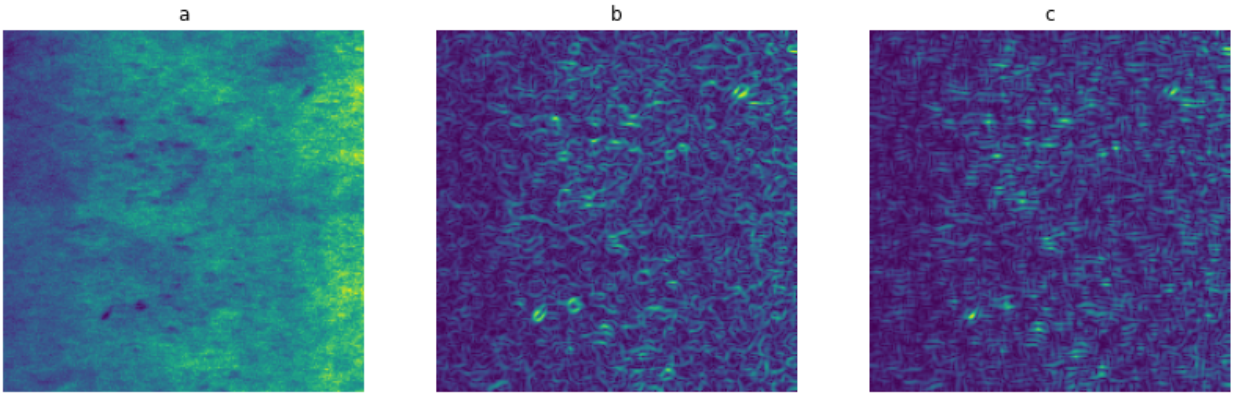

Figure 4: (a) The original image of a rolled in scale defect (b) the first derivative (c) The second derivative.

Through this method we have a method of transforming an image into a 14 element long feature vector.

2.3 Unsupervised Classification from Predetermined Image Filters

Using the feature vectors of the images found using the method described in section 2.2, we developed an unsupervised machine learning algorithm to classify the images, the classes in the first two principal components can be seen in Fig. 5.

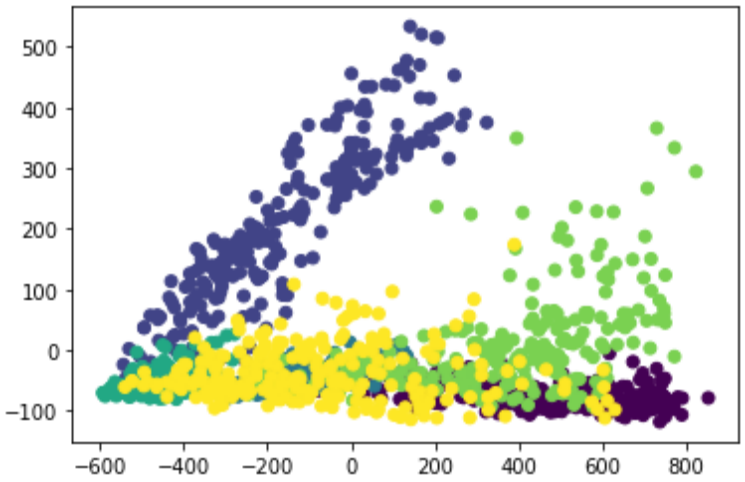

Figure. 5: Each color point represents one of the six classes of images. There is not much variation seen in the first two principal components so a more advanced feature reduction algorithm needs to be applied.

To reduce the dimensionality we applied TSNE with a perplexity of 30 to reduce it to two dimensions, shown in Figure. 6.

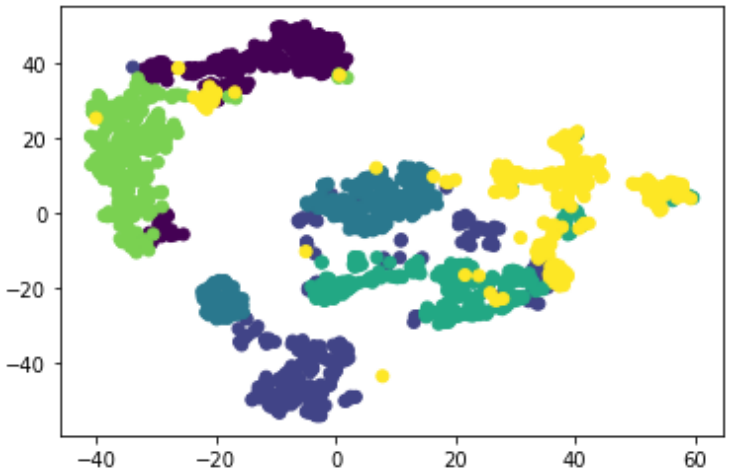

Figure. 6: The feature space after the application of TSNE.

To determine the class of each image we then applied K-Means with 6 clusters to the TSNE reduced feature space.

2.4 Supervised Classification with Random Forest

For the third method to classify the images we used the same features from section 2.2 and used a Random Forest Classifier, implemented in the scikit-learn python package,

to determine the class of each image. The classifier was trained on a balanced set of the images representing 80% of the total data and tested on the remaining 20%.

3. Results

The performance of the three different methods are shown in Table 1.

Method	**Accuracy**
CNN with log polar transform	95%
Predetermined feature vector with TSNE and K-Means	82%
Predetermined feature vector with Random Forest Classifier	97%

As we can see from Table.1 the predetermined feature vectors with TSNE and K-Means performed the worst at ~82% accuracy, however the most surprising result is that the predetermined feature vectors with Random Forest performed better than the convolutional neural network and was even competitive with previous transfer learning classifiers, this is surprising compared to its relative lack of complexity.

In Figure. 7 we can see the comparison of the confusion matrices for the three different methods.

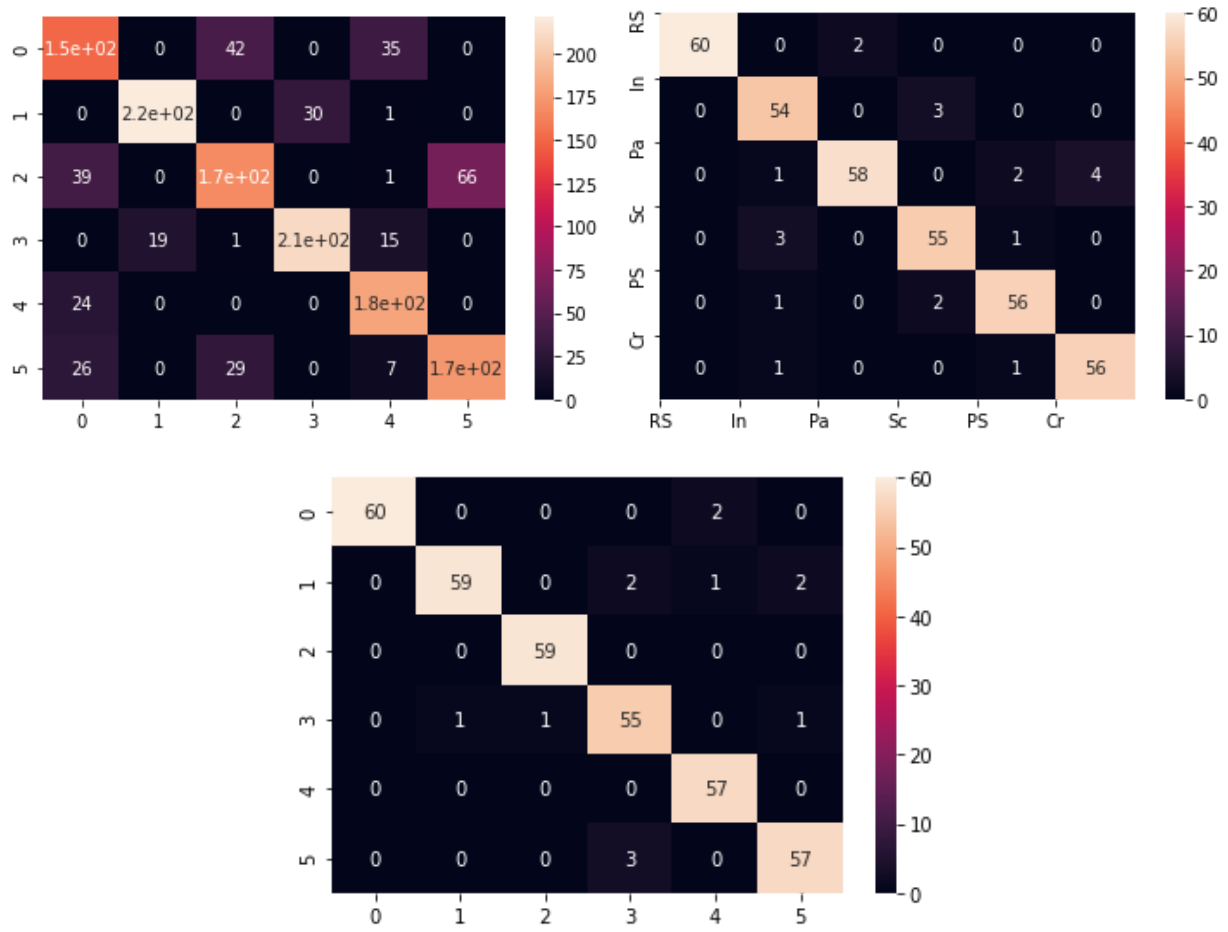

Figure. 7: (top left) TSNE/K-Means predetermined feature vector. (top right) CNN with log polar transform. (bottom) Random Forest predetermined feature vector

4. Discussion

4.1 Feature Importance

This experiment yielded some interesting results. Primarily the discovery that non-CNN methods can have competitive accuracies to CNNs was the most notable. From the Random Forest method we are able to determine which image extraction features contributed most to the classification. Below the importance of each feature is shown in Table.2

Image Feature	Importance
Unfiltered Image Standard Deviation	21%
Total Gradient Standard Deviation	17%
Total Gradient Average	16%
2nd derivative Average	13%
Unfiltered Image Average	12%
2nd derivative Standard Deviation	10%
X Gradient Average	4%
Gradient Direction Average	3%
Y Gradient Average	2%

Table 2: Feature importance from Random Forest Classifier

We see from Table 2 that the individual x and y gradients as well as the angle had little to no importance to the classification. While the primary contributors were the average and standard deviation of the 0th, 1st, and 2nd derivative of the image. To make an even simpler model we could rely wholly on those six features. Rerunning the analysis with only those features results in the same accuracy.

4.2 Goodness of fit

In Table 3 we can see the complexity of each model compared to its accuracy.

Model	Accuracy	Number of Parameters
Random Forest	0.97	2900
Transfer Learning[1]	0.99	~138 million (VGG)
CNN	0.95	5240
TSNE/K-Means	0.82	32

Table 3. Accuracies and number of parameters in each estimator, plus the estimator from [1]

It is easy to see that the RF Performs better than the CNN in both accuracy and complexity. However it is harder to determine whether the RF or the K-Means has a better goodness of fit, due to the large difference in number of parameters and accuracy.

5. Conclusions

In an environment such as a production line where high throughput and accurate detection of defects would be needed, an RF with predetermined image features would allow for much faster throughput than a CNN, due to its lower computational requirements. However, transfer learning allows higher accuracy, but slower throughput.

As we have shown, in certain instances using a CNN for small image classification tasks can be the equivalent of “using a sledgehammer to crack a nut”. A much simpler algorithm (RF with ~2900 parameters) designed for the task of identifying steel defects had accuracies 2% less than transfer learning (VGG16 with ~138 million parameters) and better than a smaller CNN (5240 parameters).

For the Jupyter Notebooks of the code used in making this paper go to:

https://github.com/nolanlad/poor-mans-CNN
https://github.com/nolanlad/NEU-CLS-FCN

References

1. Cohn, R., & Holm, E. (2021). Unsupervised Machine Learning Via Transfer Learning and k-Means Clustering to Classify Materials Image Data. Integrating Materials and Manufacturing Innovation. https://doi.org/10.1007/s40192-021-00205-8

2. Kitahara, A. R., & Holm, E. A. (2018). Microstructure Cluster Analysis with Transfer Learning and Unsupervised Learning. Integrating Materials and Manufacturing Innovation, 7(3), 148–156. https://doi.org/10.1007/s40192-018-0116-9

3. Wookeun, J. K., Kim, J. H., Lee, J. (2019). CyCNN: A Rotation Invariant CNNusing Polar Mapping and Cylindrical Convolution Layers. https://arxiv.org/pdf/2007.10588.pdf