Yousra KATEB [1], Abdelmalek KHEBLI [1], Hocine MEGLOULI[1], Salah AGUIB[3], Mohamed Salah KHELIFI TOUHAMI[2]

J. Electrical Systems 20-1 (2024): 52-67

Regular paper

Classifying Surface Fault in Steel Strips Using a Customized NasNet-Mobile CNN and Small Dataset

Journal of Electrical Systems

Steel metal is an important product in ferrous manufacturing, and the manufacturing process has to be improved so that hot-rolled strip flaws may be correctly identified. Machine-learning-based automated visual inspection (AVI) systems have been created, however they lack crucial components, such as inadequate RAM, resulting in complexity and sluggish implementation. Long execution times also result in delays or incompleteness. A scarcity of faulty samples further complicates steel defect diagnosis due to the disparity between non-defective and defective pictures. To overcome these difficulties, a deep CNN model is built using the pre-trained NasNet-Mobile backbone architecture. The model, which uses 26 times less data than other papers datasets, recognizes steel surface pictures with six faults with 99.30% accuracy, outperforming previous methods. This study is beneficial for surface fault classification when the sample size is small, the software is less effective, or time is limited. Avoiding these issues will improve safety and end product quality in the steel industry, saving time and money

Keywords: Image recognition, Steel surface, Visual Inspection, CNN, small dataset, Deep learning, Defect Classification.

I. INTRODUCTION

Hot-rolled strip steel is an important product in the steel industry [1, 2], with applications in vehicle manufacture, aerospace, and light industries. Surface quality is an important measure of market competitiveness since it varies according on raw materials, rolling process, and external environment. Imperfections such as oxide scale, inclusions, scratches, and other flaws can impair appearance and fatigue resistance[3]. However, perfecting the approach over time will not totally eliminate these flaws. Surface fault classification can be used as a reference throughout the manufacturing process to improve output and save costs.

Real-time evaluation of steel surfaces involves various problems, including hazardous locations, fast working speeds, and a wide range of surface fault types. These defects are not controlled by standards and might differ between plants and operators [43]. Hot roll coil (HRC) is the most prevalent completed steel form and a critical raw material for producers, needing accurate spot pricing and analysis. Raw material costs, worldwide trade agreements, macroeconomic indicators, and mill treatments all have an impact on pricing.

[*] Corresponding author: Yousra KATEB 1 Laboratory of Electrification of Industrial Enterprises, Faculty of Hydrocarbons and Chemistry, University Boumerdes E-mail: y.kateb@univ-boumerdes.dz
[1] Laboratory of Electrification of Industrial Enterprises, Faculty of Hydrocarbons and Chemistry, University of M'hamed Bougarra Boumerdes, City of the Independence, Boumerdes 35000, Algeria.
[2] LGPH laboratory, Faculty of Hydrocarbons and Chemistry, University of M'hamed Bougarra Boumerdes, City of the Independence, Boumerdes 35000, Algeria.
[3] Dynamic Motors and Vibroacoustic Laboratory, Faculty of Technology, University of Boumerdes35000, Algeria

Steel surface examination is now divided into two categories: traditional and deep learning approaches. Traditional approaches extract features using SVM[4], Random Forest[5], KNN [6], and GAN [44]. However, extracting characteristics from photos is difficult due to the absence of established criteria for fault distribution. Deep learning methods based on convolutional neural networks are used to classify faulty surfaces on steel products, yielding great accuracy, speed, and flexibility.

Enhancing surface defect recognition precision in hot-rolled strips can reduce the need for human involvement in defect classification, which benefits both quality inspectors and steel factories. Quality inspectors may avoid working late, which is good for their health, and mistakes caused by tiredness and other factors can be considerably reduced, improving strip steel performance and productivity [7, 8].

To summarize, enhancing surface fault classification accuracy in hot-rolled strips can result in considerable economic and societal benefits, such as decreased human intervention and increased performance and output. In brief, the paper's contributions are as follows:

- The dataset comprises about 87000 digital photos of steel flaws; however, in this study, we choose to utilize only a small number of images (300 images per class) to evaluate the efficacy of our proposed technique.
- The NasNet-Mobile CNN model is used to classify six types of faults in steel flats, with improvement strategies outlined in section 2. NasNet-Mobile was chosen because it is a basic transfer learning model with just 5.3 million parameters, making it computationally inexpensive and quick to run. In addition, it strikes a nice mix between acceptable performance and low-cost computations, making it an easy transfer learning model to utilize.
- We evaluate our method's feasibility by varying hyper-parameters. In section 3, we provide the experiments and analyze the data. Section 4 summarizes the conclusions.

II. RELATED WORK

Experts often identified issues manually, which was inaccurate and error-prone [9]. Furthermore, as a result of the same faults, different expert judgments will be generated, resulting in inaccurate types and classes of strip steel flaws and reducing defect detection reliability. Recognition findings based on researchers' subjective evaluations are typically insufficient [10, 11]. To address the limits of manual identification, academics have proposed a variety of machine learning-based alternatives.

-learning-based methodology. It trains a meta model to learn many tasks, similar to Finn's Model-Agnostic Meta-Learning algorithm (MAML) and Ravi's Long Short Term Memory network (LSTM). Existing meta-learning methods frequently incorporate an LSTM or Recurrent Neural Network (RNN) structure into the model ; nevertheless, these algorithms have high temporal complexity and slow running times. As a result, it is not suitable for industrial application.The Grayscale Covariance Matrix (GLCM) and the Discrete Shear Transform were utilized to propose a categorization strategy [14]. (DST). A GLCM computation is performed once the images have yielded multi-directional shear characteristics. It then does an important aspect analysis using high-dimensional feature vectors before being fed into a support vector machine (SVM) to detect surface flaws in strip steel. The GLCM technique's main drawback is its huge matrix dimensionality, which mandates the employment of highly competent software.

In their paper [15], the authors developed a novel multi-hyper-sphere SVM with additional information (MHSVM+) strategy for discovering hidden information in faulty data sets using an additive learning model. It offers a greater classification accuracy for faulty datasets, particularly damaged datasets. However, the SVM method underperforms in big data sets with noise and overlapping target classes. It also underperforms when features exceed training data samples.The authors [16] developed a one-class classification algorithm using generative adversarial networks (GAN) [17] and SVM. It uses characteristics generated by GANs to train an SVM classifier. It improves the loss function, increasing the model's stability. Unfortunately, the aforementioned basic Machine Learning approaches sometimes need extensive feature engineering, which significantly increases costs [18].Traditional machine learning methods, as previously stated, are typically influenced by defect size and noise. Furthermore, this method's accuracy is insufficient to meet the practical requirements of automated flaw identification. Some parts must be built by hand, and the application's scope is rather limited. Deep learning-based approaches, particularly convolutional neural networks (CNN), have shown tremendous success in image classification tasks in recent years [19, 20]. CNN has excellent characterisation capabilities [21, 22] and is particularly effective in detecting strip surface faults [6, 8, 17].

The authors [23] built on GoogLeNet [24] and refined it slightly by integrating identity mapping. To reduce overfitting, the dataset was expanded with the data augmentation method.
SqueezeNet [25] was used in the study [26] to demonstrate an end-to-end effective model. SqueezeNet now supports multiple receptive field scheduling, which might enable scale-related high-level features. It is useful for low-level feature training and can identify strip steel surface flaws quickly and reliably. One of SqueezeNet's major drawbacks is its low accuracy when compared to larger and more complex models. The authors [27] presented a modified AlexNet [28] and SVM-based intelligent surface defect detection system for hot-rolled steel strip images. Because of receptive field constraints, CNN-based classification models have good fitting capabilities but poor global representation. Obtaining a sufficient number of fault samples in complex industrial circumstances is challenging, therefore expanding the dataset has become an urgent issue that must be addressed. The attention mechanism, on the other hand, has been proven to allow the model to focus on more important information, resulting in improved recognition accuracy. In modern studies, however, attention processes are rarely utilized to characterize strip steel surface flaws [29]. Traditional machine learning approaches sometimes need extensive feature engineering, which considerably increases the cost.

III. PROPOSED APPROACH

Our plan includes four major steps. Step 1: We preprocess the data and categorize it into six types of defects: patches, crazing, pitted surface, scratches, rolling in size, and inclusion. This dataset is accessible at the SEVERSTAL Steel Detection Competition website [30].Step 2: We utilize the pre-trained CNN NasNet-Mobile as the model's backbone to extract picture features; the top layers are frozen to use the ImageNet stored weights. The last block is then completely wiped and rebuilt with a brand new one (global average pooling, dropout, exponential linear unit (ELU) to represent the dense layers, and a Softmax function for the prediction and classification layer). Step 3: We fine-tune the model using the new weights and switch between optimizers (ADAM optimizer, ADAMAX optimizer) to achieve the best results. Step 4: We do a comparison to choose the best fine-tuned model.

1. Steel Surface Defects Data set

SEVERSTAL surface defect database contains six different forms of hot-rolled steel strip surface defects: rolled-in scale (RS), inclusion (In), patches (Pa), crazing (Cr), pitted surface (PS), and scratches (Sc) [30]. The database has 1800 photographs (300 for each surface fault category). Figure 1 shows representative photographs of numerous common flaws. The dataset collection was selected because it comprises fewer photographs than other databases, enabling us to evaluate the effectiveness of our approach with such a little amount of data to datasets from other works (Table 1). The training data was created by randomly selecting 80% of the data (240 images for each fault class) from the SEVERSTAL dataset. The remaining twenty percent is utilized to confirm the network's categorization. All data was enhanced with the Tensorflow [31] and Keras [32] libraries' "Image-Data-Generator" function. Rotation (0, 45, 90, 180 degrees), horizontal flipping, shearing (0.2), and zooming (0.2). Before feeding each picture into our network, the pixel values were modified to lie within the range of [-1 ; 1].

Table 1. Summary of number of data in previous publications.

Proposed Algorithm	Image Modality	Number of Images
Deep residual neural network [22]	Severstal: Steel Defect Detection	87704
DenseNet, ResNet,U- Net [33]	Severstal: Steel Defect Detection	12568
ResNet-50, ResNet-152 [34]	Severstal: Steel Defect Detection, NEU steel database	9385
Our dataset: NasNet-Mobile	Severstal: Steel Defect Detection	1800

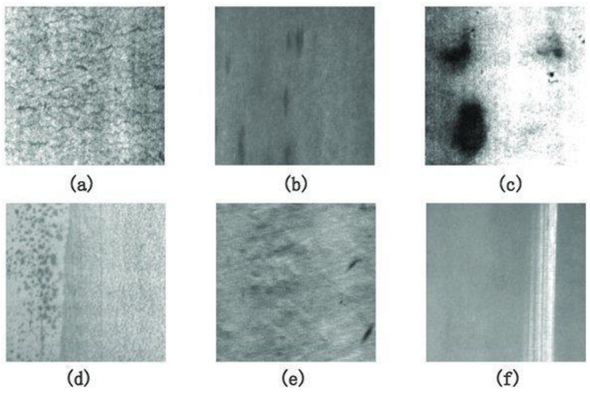

Figure 1. Several steel surface fault samples. (a) Crazing. (b) Inclusion. (c) Patches. (d) Pitted surface. (e) Rolled in scale. (f) Scratches [35].

2. Classification Model—Customized NasNet-Mobile:

Neural Architecture Search (NAS) is a technology that automates the development of neural network architecture in order to get the best results on a certain project. The goal is to create the architecture using as little resources and as little human intervention as feasible. The NasNet architecture, devised by the authors of [36], is a neural architecture search network that trains to acquire the most correct parameters from the produced architecture using a recurrent neural network (RNN) and reinforcement learning. Designing a CNN architecture takes a long time when the data is huge, such as the ImageNet dataset. They then created a CNN framework capable of finding the optimal architecture in a small amount of data and then transferring that architecture to be trained on large datasets; this architecture is known as "learning transferable architectures". The NasNet-Mobile architecture can be scaled according to data volume.

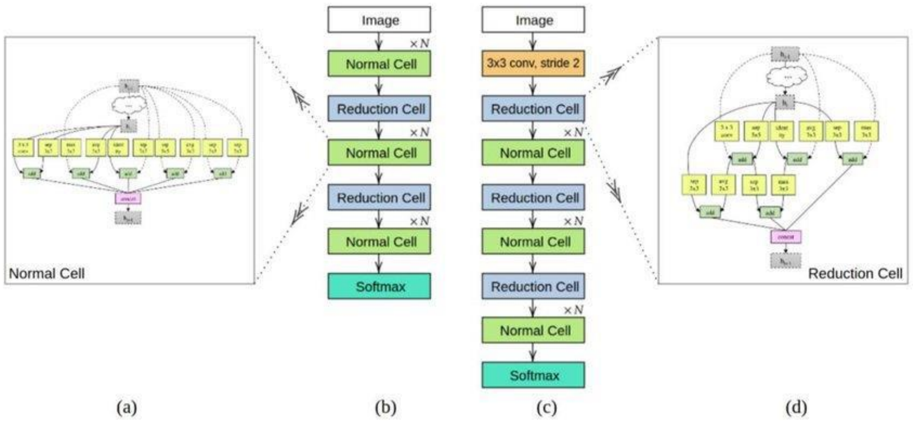

Figure 2. The NasNet was scaled for use with (b) the CIFAR10 database and (c) the ImageNet dataset, as well as examples of an ordinary cell (a) and a reduction cell (d). [36]

a. Depthwise and pointwise convolutions:

The NasNet-Mobile framework is built on depthwise separable convolutions [37], a type of factorized convolution in which a standard convolution is split into a depthwise convolution and a 1 x 1 convolution defined as a pointwise convolution. NasNet-Mobile use depthwise convolution to apply a separate filter to each input channel. The pointwise convolution then performs a 1 x 1 convolution on the depthwise convolution outputs. A traditional convolution algorithm filters and mixes inputs in a single step to produce a new set of outputs. The depth-wise separable convolution divides this into two layers: filtering and combining. This factorization greatly decreases processing and model size. Depth-wise separable convolutions are composed of two layers: depthwise and pointwise. We employ depth-wise convolutions (input depth) to create a single filter for each input channel. The depthwise layer output is then linearly combined using pointwise convolution, which is a simple 1 x 1 convolution (Eq. (1)).

$$G_{k,l,m} = \sum_{i,j} \hat{k}_{i,j} * F_{k+i-1,\ i+j-1,\ m} \qquad (1)$$

Here, K is the depth-wise convolutional kernel with a size of $Dk\ x\ Dk\ x\ M$ Where the m^{th} filter in $\hat{k}$ is applied to the m^{th} channel in F to output the m^{th} channel of the filtered output feature map G. As shown in (Figure 2), RNN merges two hidden layers to move on to the following hidden layer.

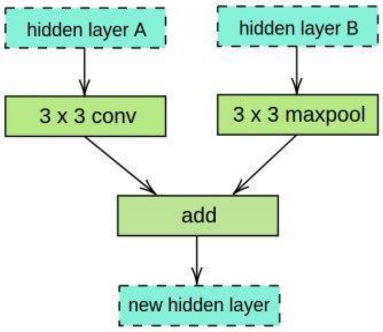

Figure 3. Convolution cell block obtained from RNN search.

We investigate changes in the architectural design of each reference structure experimentally (Section II). To forecast the steel defect class, we use transfer learning from neural network models trained on ImageNet [38] in the simplified CNN framework, removing the original block and replacing it with a new block with global average pooling, dropout, dense, and a Softmax function for the final prediction layer (Figure 3). The whole architecture is frozen for the initial phase of training (before fine-tuning), with the exception of the final produced block. Following that, we unfreeze the model's top, allowing it to train anew toward the intended objective (steel defect classification). This prevents the network from overfitting during training and lets the model to learn faster and for a longer length of time, resulting in better generalization. Using the light-weight NasNet architecture has several advantages, including enhanced model training, reduced susceptibility to short dataset overfitting, and portability to different embedded systems.

b. NasNet-Mobile for Fault Classification:

There are three primary reasons for using this CNN as a cornerstone of our model. The model is lightweight, using just 23 MB of RAM compared to comparable models such as VGG16 (549 MB), ResNet52 (232 MB), and NASNetLarge (343 MB), among others. Second, in terms of parameter count, this model has just 5.3 million parameters, which is quite modest (for comparison, the VGG16 has 143.7 million parameters, making it 27 times bigger than our NasNet-Mobile model). The last argument is that this model requires just 27 ms per inference step in a CPU and 6.7 ms per inference step on a GPU, which is 60 times less than the EfficientNetB7 (1578.9 ms per inference step).

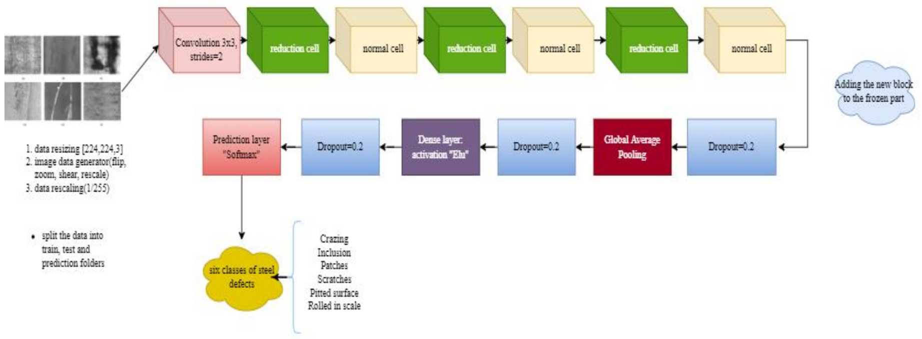
Convolution 3x3, strides=2
reduction cell
normal cell
reduction cell
normal cell
reduction cell
normal cell
Adding the new block to the frozen part
Dropout=0.2
Global Average Pooling
Dropout=0.2
Dense layer: activation "Elu"
Dropout=0.2
Prediction layer "Softmax"
1. data resizing [224,224,3]
2. image data generator(flip, zoom, shear, rescale)
3. data rescaling(1/255)
split the data into train, test and prediction folders
six classes of steel defects
Crazing
Inclusion
Patches
Scratches
Pitted surface
Rolled in scale

Figure 4. General framework of the suggested approach.

c. Improved NasNet-Mobile :

The basic NasNet-Mobile model is pre-trained with 1,056 ImageNet [38] recognition output channels. The main exploration in this design revolves upon the number of regular cells in the model. We used three reduction cells and three standard cells in our modified NasNet-Mobile architecture. The total amount of parameters is 4,376,022, with only 106,306 (2.42%) trainable and the rest being frozen. We develop the backbone using the pre-trained NasNet-Mobile architecture, which consists of six cells (reduced and normal), followed by a freshly designed defect classification block that contains a convolution layer, dropout, dense, and global average pooling. The first dense layer uses "ELU" instead of "ReLU" as the activation function. ELU, or Exponential Linear Unit, is a function that converges cost to zero quicker and with higher accuracy [39]. In contrast to other activation functions, ELU has an additional alpha constant that must be positive, as shown in Eq. (2).

$$R(z) = \begin{cases} z & when\, z > 0 \\ \alpha.(\exp(z) - 1) & when\;\; z < 0 \end{cases} \tag{2}$$

ELU is quite similar to RELU, except for the negative inputs. They are both in the identity function form for non-negative inputs. In contrast, ELU smoothes gradually until it reaches -α, whereas RELU smoothes significantly (see Figure 4). ELU is preferred over ReLU as an activation function because it smoothes out gradually till reaching α, while RELU smoothes out rapidly. In addition, unlike ReLU, ELU can generate negative outputs.The ELU is a continuous and differentiable activation function that provides quicker training durations than other linear non-saturating functions, such as ReLU and its several variants (Leaky-ReLU (LReLU) and Parameterized-ReLU). It does not suffer from dying neurons, exploding or disappearing gradients, and achieves higher accuracy than other activation functions like as ReLU, Sigmoid, and Hyperbolic Tangent. Steel surface fault classifier parameters can be updated by lowering a multi-class loss function referred to as Categorical cross-entropy (Eq. (3)).

$$LOSS = -\sum_{i=1}^{output\ size} y_i * \log \hat{y}_i \qquad (3)$$

The model output is represented by yi (ith scalar value), $\hat{y}_i$ (equivalent target value), and output size (number of scalar values).

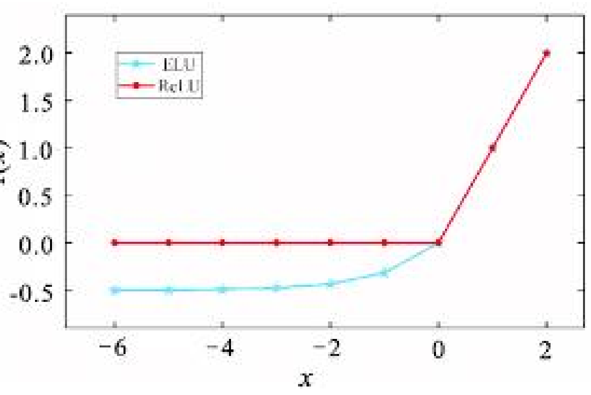

Figure 5. Graph demonstrating the distinction between ELU (green) and ReLU (red) activation functions [39].

d. Optimized model for steel surface defects :

Following the creation of the fundamental NasNet-Mobile model for steel surface defect detection, we present a number of possible solutions for increasing accuracy and lowering execution time. First, data augmentation is utilized to increase the number of characteristics that the model learns. Second, a new block, which we have previously specified, is added to the bottom of the model for the prediction component ; this block will aid enhance accuracy while reducing model parameters and execution time. Third, we try different optimizers to see which one works best (ADAM and ADAMAX). Finally, we lower the learning rate using exponential decay as described in (Eq. (4)), and we apply early halting when the model accuracy cannot be improved further. The model recovers the optimal weights.

$$y = a(1-b)^x \qquad (4)$$

where, y defines the final value, a is the initial value, b represents the decay factor, in addition to x refers to the value of time that has passed.

IV. EXPERIMENTS AND RESULTS

Our technique is built on the publicly accessible Python framework from Google Colaboratory [40]. Tesnorflow [31], Keras, Matplotlib, NumPy, and Glob are the primary libraries utilized in this implementation. We used 80% of the photographs from the Severstal (Severstal) collection as training data (240 images for each fault category) and 20% as validation data. Performance is measured both before and after the network has been fine-tuned. Table 2 displays the values of the hyper-parameters used to train the CNN. The experiments were carried out with Windows 10 Professional on the Intel® Core(TM) - i5 7200U, 64-bit platform with 8GB RAM and NVIDIA RTX 2070, taking use of the free accessible GPU on Google Colab Platform.

The training using the surface defect data was really quick. The model was trained in 3414 seconds (56 minutes and 54 seconds) before fine-tuning, and in 528 seconds (8 minutes and 48 seconds) after.

Table 2. Hyperparameters for training NasNet-Mobile convolutional neural network.

Hyperparameters	Before Fine-Tuning	After Fine-Tuning
Number of epochs	15	110
Steps per epoch	6	6
Number of trainable parameters	106,306	4,376,022
Learning rate mode	Max	Exponential decay
Restore best weights	True	True
Loss	Categorical Crossentropy	Categorical crossentropy
Optimizer	ADAM	ADAMAX

Table 3. Metrics of performance for NasNet-Mobile in both training and validation datasets before to fine-tuning.

Metrics	Training Dataset	Validation Dataset
Accuracy	99.30%	97.78%
Loss	0.029	0.063
Precision	99.50%	98.61%
Recall	99.50%	97.77%
Area under the curve AUC	100%	99.95%

1. Model evaluation before fine-tuning:

The model was evaluated twice, first without fine-tuning parameters and once with them, using deep learning measures including accuracy, loss, recall, AUC, FP, FN, TP, TN, and precision. Both the training and validation datasets yielded good results. However, there was a minor loss in performance due to model learning from training data, which improved reliability. The validation data, which comprises just 20% of the entire data, has not been learnt from. To guarantee fair evaluation, the model was evaluated via held-out sampling. Fine-tuning and dataset augmentation are two strategies for mitigating performance discrepancies, ensuring that the model can learn new characteristics. According to the findings, adding more elements to the model might increase its performance significantly.

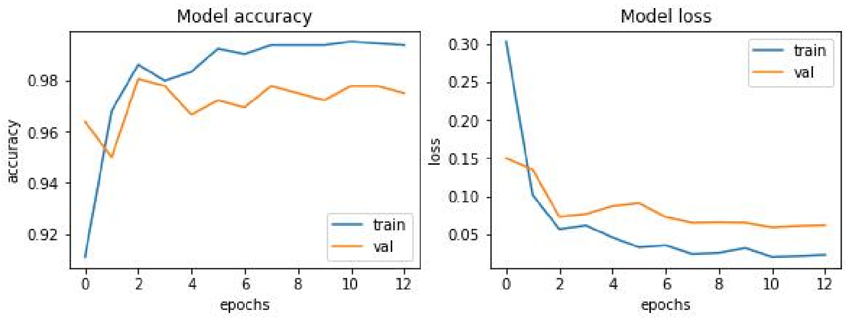

Figure 6. Accuracy and loss before fine-tuning the NasNet-Mobile

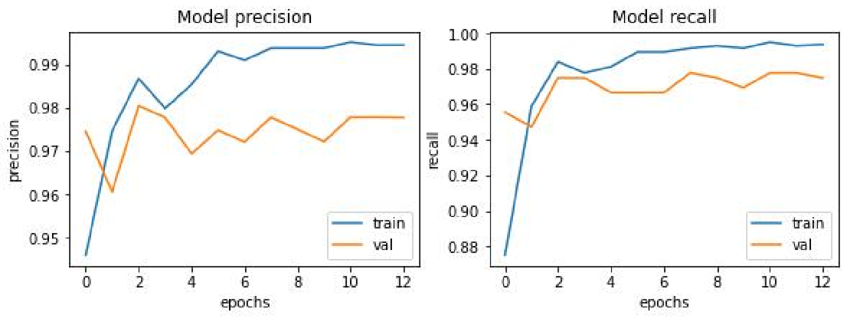

Figure 7. Precision and accuracy before fine-tuning the NasNet-Mobile.

NasNet-Mobile was optimized using a dataset of 1800 photographs augmented using Image Data Generator. The training lasted 20 epochs as shown in figure 6, and the training loss decreased as the number of epochs rose. During the learning phase, the model determined the visual prominence of the reference and candidate images, minimizing training loss. During the testing phase, photos from a separate class were chosen at random and given to the network for the first time during training. As training continued, the accuracy of the test set matched that of the training set, with just a tiny difference. This shows that the algorithm successfully anticipates instances that are not in the training set.Precision is defined as the proportion of correctly identified examples (5), whereas recall (also known as sensitivity) is the proportion of retrieved relevant cases (6). Relevance thus governs precision and recall.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{5}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{6}$$

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{7}$$

As seen in the prior curves (Figure 8) and (Table 3), the best-achieved accuracy is around 99.51% in training and 98.6% in validation. The recall is around 99.30% for training and 97.78% for validation datasets. These findings were obtained prior to fine-tuning.

2. Model evaluation before fine-tuning:

Table 4. Performance of the model after fine-tuning.

Metrics	Training dataset	Validation dataset
Accuracy	100%	98.06%
Loss	0.0243	0.0728
Precision	100%	98.03%
Recall	100%	97.51%
Area under the curve AUC	100%	99.41%

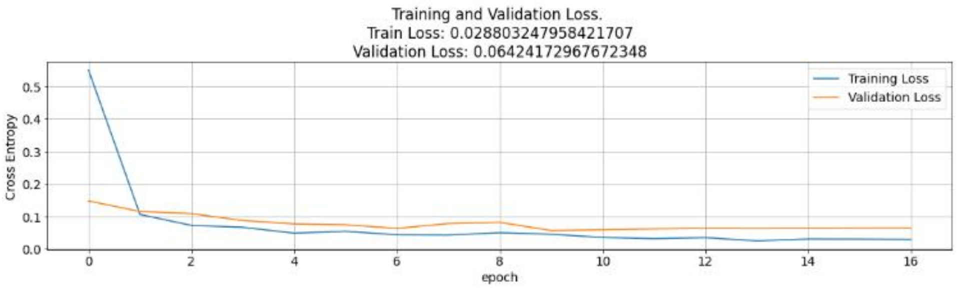

Figure 8. Training and validation loss.

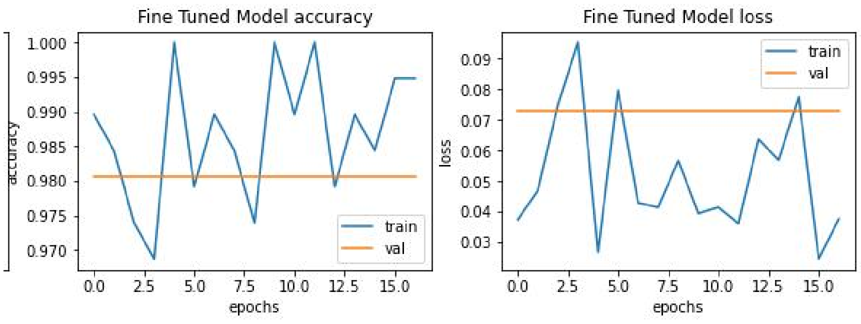

Figure 9. Accuracy (left) with loss (right) curves of the fine-tuned NasNet-Mobile.

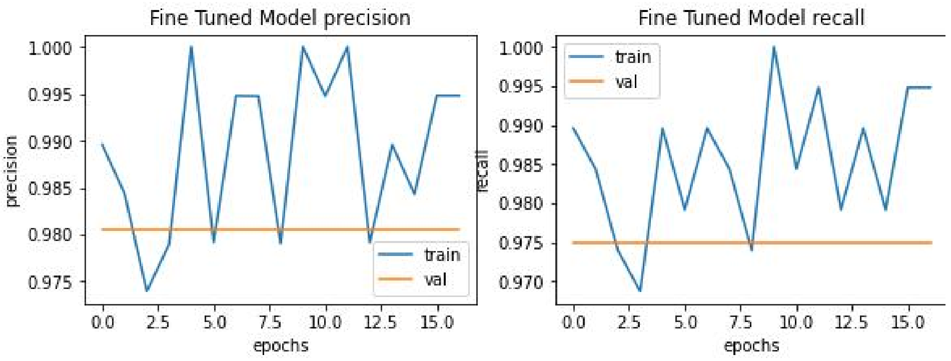

Figure 10. Precision with recall in the fine-tuned Model.

The training procedure was terminated prematurely due to no progress in accuracy and loss across three consecutive epochs. This instability is due to the lack of batch normalization in the trials, which improves the training process by lowering internal covariate changes, increasing stability, and optimizing the model. Batch normalization increases generalization by normalizing layer activations, which reduces overfitting and initial weight sensitivity. To address this issue, model fine-tuning should be performed by adding different batch normalization layers and retraining the model.

V. COMPARATIVE STUDY

The study compares a model for steel surface inspection using various methodologies, considering accuracy, executing time, model lightness, and data size. The model achieves a higher accuracy rate of 99.51% compared to other models. The NasNet-Mobile model is lighter and requires less data, with 5.3 million parameters compared to DeCAF and residual neural network models. The network runs faster than other models, with a three-fold lower executing time and 2.5-fold lower executing time. The model's error rate is the lowest, with a 0.029 error rate, 22 times less than CNN methods with LU's activation function. This suggests the model can learn better with less errors and better accuracy.

Table 5. The classification accuracy (%) for some state-of-the-art steel surface fault classification algorithms taking into consideration the triplet model lightness vs. running time vs. data size.

Method	Accuracy	Model Lightness (nbr of prmtrs)	Time (ms) Per Inference step	Data Size
[34]	96.91%	25.6 million	58.2	87704
[33]	98.56%	25.6 million	77.1	12568
[41]	97.27%	60 million	10.3	26334
Our method (Developed NasNet-Mobile)	99.30%	5.31 million	27.5	1800

Table 6. Evaluation of the classification errors depending on several activation functions, the model used the name of the activation function wrapped into parenthesis.

Method	Error Rate (%)
CNNs (Sigmoid)	12.8721
CNNs (Hyperbolic Tangent)	18.1129
CNNs (ReLUs) Recall	9.5018
Our method (ELUs)	**0.029**

VI. CONCLUSIONS

In this study, we proposed a new approach for classifying flaws in steel sheets using a pre-trained NasNet-Mobile CNN. The following are the key conclusions of the research:

The issue of accurate classification when memory is limited is addressed with the NasNet-Mobile network, which contains less parameters than other CNNs (5.3 million parameters). The top layers of this CNN were frozen, which allowed for less memory and calculations while maintaining weights. The problem of long execution times is solved by utilizing Google Colab Platform's free accessible GPU.

The problem of dataset shortage is addressed in this study and overcome owing to the potency of this CNN, which can acquire the essential features of the picture due to its large depth (389 layers), which can extract more features even if the dataset quantity is little.

When hyperparameters were fine-tuned, the model improved. We discovered that the ADAMAX optimizer outperformed the ADAM optimizer in our modified NasNet-Mobile architecture. As a result, we saw a small improvement in both accuracy and error rate. The Adam optimizer modifies weights in inverse proportion to the scaled L2 norm of prior gradients, whereas AdaMax extends this to the infinite norm of previous gradients.

The model has been rationalized, and the last block has been completely eliminated and replaced with a new one. As a result, various alterations were made to meet the needs of the steel image characteristics. The convolution layers included an ELU activation function. We intended to apply this activation function in order to benefit from its advantages. This activation function aids in the solution of the dying RELU issue, which occurs when the gradient value on the graph's negative side is 0. As a result, the weights and biases of specific neurons do not change during the backpropagation process. This can lead to dead neurons that never fire. The RELU dying problem can be addressed by incorporating a log curve for negative input values. It then helps the network to modify weights and biases in the right direction. The ELU activation function improves model performance and network robustness. It progressively smoothes until the output equals -α, while RELU smoothes substantially.

A dropout (0.2) was introduced after each fully connected layer, and a global average pooling is put before the dense layer, which helped to minimize time and memory throughout the model construction. Learning rate scheduling allows us to employ bigger steps for the first few epochs before progressively reducing the number of steps as the weights approach their optimum value.

More improvements can be accomplished by obtaining more training data, strengthening the network's architecture, and fine-tuning its hyper-parameters rather than increasing the training epochs under the current structure, which may result in overtraining. In overall, we can infer that the proposed approach may be used in image processing tasks such as classification to solve the three major challenges: time consumption, memory inadequacy, and limited data encountered in small factories and less-powerful operators.

VII. REFERENCES

[1] Yi, L., Li, G., Jiang, M. (2017). An end-to-end steel strip surface defects recognition system based on convolutional neural networks. Steel Research International, 88(2): 1600068. https://doi.org/10.1002/srin.201600068

[2] Kateb, Y., Meglouli, H., Khebli, A. (2020). Steel surface defect detection using convolutional neural network. Algerian Journal of Signals and Systems, 5(4): 203-208. https://doi.org/10.51485/ajss.v5i4.122

[3] Neogi, N., Mohanta, D.K., Dutta, P.K. (2014). Review of vision-based steel surface inspection systems. EURASIP Journal on Image and Video Processing, 2014(1): 1-19. https://doi.org/10.1186/1687-5281-2014-50

[4] Zhang, M., Zheng, H.F., Hao, N.I. (2019). Ultrasonic image defect classification based on support vector machine optimized by genetic algorithm. Acta Metrologica Sinica, 40: 887-892.

[5] Du, P., Samat, A., Waske, B., Liu, S., Li, Z. (2015). Random forest and rotation forest for fully polarized SAR image classification using polarimetric and spatial features. ISPRS Journal of Photogrammetry and Remote Sensing, 105: 38-53. https://doi.org/10.1016/j.isprsjprs.2015.03.002

[6] Kim, C., Choi, S., Kim, G., Joo, W. (2006). Classification of surface defect on steel strip by KNN classifier. Journal of the Korean Society for Precision Engineering, 23(8): 80-88.

[7] Wang, J., Luo, L., Ye, W., Zhu, S. (2020). A defect-detection method of split pins in the catenary fastening devices of high-speed railway based on deep learning. IEEE Transactions on Instrumentation and Measurement, 69(12): 9517-9525. https://doi.org/10.1109/TIM.2020.3006324

[8] Li, Z., Tian, X., Liu, X., Liu, Y., Shi, X. (2022). A two-stage industrial defect detection framework based on improved-YOLOv5 and optimized-inception-resnetv2 models. Applied Sciences, 12(2): 834. https://doi.org/10.3390/app12020834

[9] Vannocci, M., Ritacco, A., Castellano, A., Galli, F., Vannucci, M., Iannino, V., Colla, V. (2019). Flatness defect detection and classification in hot rolled steel strips using convolutional neural networks. In: Rojas, I., Joya, G., Catala, A. (eds) Advances in Computational Intelligence. IWANN 2019. Lecture Notes in Computer Science(), vol 11507. Springer, Cham. https://doi.org/10.1007/978-3-030-20518-8_19

[10] Wu, C., Ju, B., Wu, Y., Lin, X., Xiong, N., Xu, G., Li, H., Liang, X. (2019). UAV autonomous target search based on deep reinforcement learning in complex disaster scene. IEEE Access, 7: 117227-117245. https://doi.org/10.1109/ACCESS.2019.2933002

[11] Luo, Q., He, Y. (2016). A cost-effective and automatic surface defect inspection system for hot-rolled flat steel. Robotics and Computer-Integrated Manufacturing, 38: 16-30. https://doi.org/10.1016/j.rcim.2015.09.008

[12] Finn, C., Abbeel, P., Levine, S. (2017). Model-agnostic meta-learning for fast adaptation of deep networks. In International Conference on Machine Learning, pp. 1126-1135.

[13] Ravi, S., Larochelle, H. (2016). Optimization as a model for few-shot learning. In International conference on learning representations. International Conference on Learning Representations.

[14] Ashour, M.W., Khalid, F., Abdul Halin, A., Abdullah, L.N., Darwish, S.H. (2019). Surface

defects classification of hot-rolled steel strips using multi-directional shearlet features. Arabian Journal for Science and Engineering, 44: 2925-2932. https://doi.org/10.1007/s13369-018-3329-5
[15] Gong, R., Wu, C., Chu, M. (2018). Steel surface defect classification using multiple hyper-spheres support vector machine with additional information. Chemometrics and Intelligent Laboratory Systems, 172: 109-117. https://doi.org/10.1016/j.chemolab.2017.11.018
[16] Liu, K., Li, A., Wen, X., Chen, H., Yang, P. (2019). Steel surface defect detection using GAN and one-class classifier. 2019 25th International Conference on Automation and Computing (ICAC), Lancaster, UK, pp. 1-6. https://doi.org/10.23919/IConAC.2019.8895110
[17] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y. (2020). Generative adversarial networks. Communications of the ACM, 63(11): 139-144. https://doi.org/10.1145/3422622
[18] Li, S., Wu, C., Xiong, N. (2022). Hybrid architecture based on CNN and transformer for strip steel surface defect classification. Electronics,11(8): 1200. https://doi.org/10.3390/electronics11081200
[19] Khebli, A., Meglouli, H. (2021). Représentations d'images pour la recherche et la classification d'images. Doctoral dissertation, Automatique Appliquée et Traitement du Signal, Université M'hamed Bougara - Boumerdes Algeria.
[20] Kateb, Y., Meglouli, H., Khebli, A. (2023). Coronavirus diagnosis based on chest X-ray images and pre-trained DenseNet-121. Revue d'Intelligence Artificielle, 37(1): 23-28. https://doi.org/10.18280/ria.370104
[21] Khebli, A ., Meglouli, H., Bentabet, L., Airouche, M. (2019). A new technique based on 3D convolutional neural networks and filtering optical flow maps for action classification in infrared video. Journal of Control Engineering and Applied Informatics, 21(4): 43-50.
[22] Konovalenko, I., Maruschak, P., Brezinová, J., Viňáš, J., Brezina, J. (2020). Steel surface defect classification using deep residual neural network. Metals, 10(6): 846. https://doi.org/10.3390/met10060846
[23] Liu, Y., Geng, J., Su, Z., Zhang, W., Li, J. (2019). Real-time classification of steel strip surface defects based on deep CNNs. In: Jia, Y., Du, J., Zhang, W. (eds) Proceedings of 2018 Chinese Intelligent Systems Conference. Lecture Notes in Electrical Engineering, vol 529. Springer, Singapore. https://doi.org/10.1007/978-981-13-2291-4_26
[24] Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A. (2015). Going deeper with convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1-9.
[25] Iandola, F.N., Han, S., Moskewicz, M.W., Ashraf, K., Dally, W.J., Keutzer, K. (2016). SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and< 0.5 MB model size. arXiv preprint arXiv:1602.07360. https://doi.org/10.48550/arXiv.1602.07360
[26] Fu, G., Sun, P., Zhu, W., Yang, J., Cao, Y., Yang, M.Y., Cao, Y. (2019). A deep-learning-based approach for fast and robust steel surface defects classification. Optics and Lasers in Engineering, 121: 397-405. https://doi.org/10.1016/j.optlaseng.2019.05.005
[27] Boudiaf, A., Benlahmidi, S., Harrar, K., Zaghdoudi, R. (2022). Classification of surface defects on steel strip images using convolution neural network and support vector machine. Journal of Failure Analysis and Prevention, 22(2): 531-541. https://doi.org/10.1007/s11668-022-01344-6
[28] Krizhevsky, A., Sutskever, I., Hinton, G.E. (2012). Imagenet classification with deep convolutional neural networks. Communications of the ACM, 60(6): 84-90. https://doi.org/10.1145/3065386
[29] Hao, Z., Li, Z., Ren, F., Lv, S., Ni, H. (2022). Strip steel surface defects classification based on generative adversarial network and attention mechanism. Metals, 12(2): 311. https://doi.org/10.3390/met12020311
[30] B. Alexey Grishin, iBardintsev, inversion, Oleg. Severstal: Steel Defect Detection, 2019. Available online: https://kaggle.com/competitions/severstal-steel-defect-detection.
[31] Girija, S.S. (2016). Tensorflow: Large-scale machine learning on heterogeneous distributed systems. Software, 39(9). Available: tensorflow.org.
[32] Keras. GitHub. (2015). GitHub. https://github.com/fchollet/keras.
[33] Damacharla, P., Rao, A., Ringenberg, J., Javaid, A.Y. (2021). TLU-net: A deep learning approach for automatic steel surface defect detection. In 2021 International Conference on Applied Artificial Intelligence (ICAPAI), Halden, Norway, pp. 1-6. https://doi.org/10.1109/ICAPAI49758.2021.9462060

[34] Konovalenko, I., Maruschak, P., Brevus, V., Prentkovskis, O. (2021). Recognition of scratches and abrasions on metal surfaces using a classifier based on a convolutional neural network. Metals, 11(4): 549. https://doi.org/10.3390/met11040549

[35] Lv, X., Duan, F., Jiang, J.J., Fu, X., Gan, L. (2020). Deep metallic surface defect detection: The new benchmark and detection network. Sensors, 20(6): 1562. https://doi.org/10.3390/s20061562

[36] Zoph, B., Vasudevan, V., Shlens, J., Le, Q.V. (2018). Learning transferable architectures for scalable image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, pp. 8697-8710. https://doi.org/10.1109/CVPR.2018.00907

[37] Howard, A.G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., Adam, H. (2017). Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861. https://doi.org/10.48550/arXiv.1704.04861

[38] Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In 2009 IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, USA, pp. 248-255. https://doi.org/10.1109/CVPR.2009.5206848

[39] Ide, H., Kurita, T. (2017). Improvement of learning for CNN with ReLU activation by sparse regularization. In 2017 International Joint Conference on Neural Networks (IJCNN), Anchorage, AK, USA, pp. 2684-2691. https://doi.org/10.1109/IJCNN.2017.7966185

[40] Van Rossum, G., Drake, F.L. (1995). Python Reference Manual (Vol. 111, pp. 1-52). Amsterdam: Centrum voor Wiskunde en Informatica.

[41] Ren, R., Hung, T., Tan, K.C. (2017). A generic deep-learning-based approach for automated surface inspection. IEEE Transactions on Cybernetics, 48(3): 929-940. https://doi.org/10.1109/TCYB.2017.2668395

[42] I. Konovalenko, P. Maruschak, V. Brevus, and O. Prentkovskis, "Recognition of Scratches and Abrasions on Metal Surfaces Using a Classifier Based on a Convolutional Neural Network," Metals, vol. 11, no. 4, p. 549, 2021-03-28 2021, doi: 10.3390/met11040549. URL: https://doi.org/10.3390/met11040549

[43] R. Chemes Eddine and S. Bouras, "Detection of bearing defects using Hilbert envelope analysis and fast kurtogram demodulation method," Journal of Electrical Systems, vol. 16, pp. 92-104, 11/23 2019.

[44] L. Ahcene, A. Guerroui, T. Bordjiba, and A. Moussaoui, "Neural Networks Prediction of Ionic Mobilities in SF6-N2 mixture," Journal of Electrical Systems, vol. 14, pp. 86-94, 03/01 2018.

[45] B. Tahar and H. Merabet, "Vibration for detection and diagnosis bearing faults using adaptive neurofuzzy inference system," Journal of Electrical Systems, vol. 14, pp. 95-104, 03/01 2018.