2021 IEEE 17th International Conference on Automation Science and Engineering (CASE)
August 23-27, 2021. Lyon, France

Real-time Steel Surface Defect Recognition Based on CNN

Anna Litvintseva,[1] Oleg Evstafev[1] and Sergey Shavetov[1]

2021 IEEE 17th International Conference on Automation Science and Engineering (CASE) | 978-1-6654-1873-7/21/$31.00 ©2021 IEEE | DOI: 10.1109/CASE49439.2021.9551414

Abstract— **Steel is one of the most important building materials of our time, and the process of producing flat plates is complex. Before steel is shipped or delivered, the sheets must undergo a thorough inspection procedure to avoid defects. Identification and classification of rolled metal surface defects are one of the main tasks for the correct evaluation of product quality. This work aims to develop a method for recognizing and classifying defects of metal surfaces by their images in real-time. The algorithm is aimed at improving production standards and process efficiency. In this paper, Deep Learning (DL) and Computer Vision (CV) techniques are used to solve the problem of defect detection on the steel surface sheets. Convolutional Neural Network (CNN) architectures are compared, and various steel defects are detected and recognized. The result of this work is a comparative analysis of DL models and the choice of an algorithm designed to search and classify defects in real-time. The use of one CNN model can make it possible to create a tool that greatly facilitates the work of a person.**

I. INTRODUCTION

Surface defects in steel strips lead to deterioration of their quality. Identification and classification of rolled steel surface defects is one of the main tasks for proper evaluation of product quality. Historically, these tasks have been performed by humans. However, due to many production factors, such as high rolling speed and metal temperature, the results of such human labor are rather low [1]. It is important to detect defects in real time, this allows us to stop the metal rolling process in time and reduce the amount of rejected material.

To solve the problem of inspection of cold rolled steel, i.e. non-destructive flaw detection use different methods of non-contact quality control of sheet metal products such as: eddy current systems, ultrasonic diagnostics, using magnets, laser methods and methods of visual methods using different cameras [2]. Each of these methods can provide video inspection of cold rolled steel to a different degree. The use of cameras allows to apply classical methods and algorithms of computer vision, i.e. algorithms of image segmentation, methods of shape and color measurement, application of Sobel filter, as well as methods using machine learning, neural networks and various combinations of these methods.

In [3] a method of identifying surface defects in rolled metal products by video images is considered. Based on the image obtained by the camera with low shutter speed, a brightness threshold value is calculated, by which the background is separated from the defect area. Based on the obtained defect area, the defect features are calculated, which are fed to the input of the neural network. The trained neural network determines the defect class. The disadvantage of the Fourier image analysis is the lack of insight into the local properties of the signal with rapid temporal changes in its spectral composition, which complicates the work of the recognizer in the production process.

In [4] the authors consider a means of identification based on the Gabor filter with subsequent analysis of the distribution histogram by the results of the filter application. The main disadvantage of this tool is the limitation on the rolling speed up to 3.5 m/s.

In [5] a means of identifying surface defects using digital video cameras. The main idea of the method is that an image is formed by means of spectral illumination directed at the surface. The color image obtained after surface scanning is converted into an image with zero contrast, then the image is normalized and binarised. The identification of surface defects is done automatically by an Artificial Neural Network (ANN). If the neural network fails to identify the class of surface defect with a given accuracy, an automatic start-up of a dynamic production-situation expert system is performed, which recognizes the defect class and estimates its parameters according to genetic and morphological features. The neural network identifies 3 classes of defects in rolled steel rails: a foil, a rolled crack and a rolled bubble. The disadvantage of the method is the lack of recognition of defects typical for thin sheet rolled non-ferrous metals: hole, dents, rust spots, roll imprints and small waves. In addition, in the prototype features of the identification are formed throughout the image, which significantly increases the processing time of one frame, and use of matrix cameras with a shooting speed of 25-30 FPS (frames per second) and the rolling speed of 6 m/s leads to a ”blurring” of the image, which makes it impossible to identify defects in the production process with the required accuracy.

The author's team has implemented a method for identifying surface defects [6], based on the use of a wavelet transform of rolling images with threshold filtering. The method makes it possible to detect surface defects for their elimination during reverse runs. However, this method doesn't allow determining the defect class and, respectively, limits the possibility of forming control actions by the rolling mill operator [7].

A study of the application of Convolutional Neural Networks (CNN) [8] showed that there is a problem with the correct classification of small defects, which requires separate attention of the researcher. Using a modified YOLO (You Only Look Once) architecture it was possible to achieve high quality performance at 83 FPS with an input image size of 300×300 pixels [9], but this resolution is not enough to detect small defects (0.25 - 0.5 mm) across the width of the

[1] Anna Litvintseva, Oleg Evstafev and Sergey Shavetov are with Faculty of Control Systems and Robotics, ITMO University, Kronverksky av., 49, 197101, Saint-Petersburg, Russia, oaevstafev@itmo.ru

rolling.

The development of surface monitoring tools is an important task [10]. Neural network based methodology has proven to be effective in various industries [11],[12], providing high speed, reliability, and accuracy. So we have chosen for comparison segmentation models that allow you to simply combine detection and classification tasks. This simplifies the integration of the system into any software package. The aim of our work is to compare chosen models and select an algorithm for localizing and classification steel surface defects to improve the production standards and process efficiency.

Section I gives an introduction and explains the current technology for steel detection. In Section II we present general information about the methods and data used in the work. Some experimental results are presented in Section III. Finally, Section IV — conclusion and summary of our work.

II. METHODS

A. Defect classification

The various flat surface defects in steel are usually caused by mechanical or metallurgical defects in industrial production. It is known that defects in rolled steel are standardised and can be classified into different types. At the same time, modern systems classify defects according to the description of their parameters, which may differ depending on the technological conditions of rolling [13],[14]. They are roller marks, longitudina scratches, horizontal scratches, inclusions, scars, holes, waves, pitting, air bubbles, flaking, water droplets, convex bags, meshes, star cracks, foreign bodies, heavy skin, wrinkles and longitudinal cracks respectively. In modern metallurgical production, the number of defects is much less if the technological conditions of rolling are observed. These include films, cracks, mechanical defects and holes. Within the same class, defects can be coloured by shape, appearance and structure, which makes their classification more difficult.

B. Dataset

An open dataset presented on Kaggle in the competition Severstal: Steel Defect Detection was chosen for training. The dataset contains 12,568 annotated images with four defect classes, some images contain several defect classes, and some do not contain defects at all. Training images are 1600×256 pixels in size. Figure 1 shows an example of an image from a dataset with a mask.

In total, the training sample contains 897 images with class 1 damage, 247 images with class 2 damage, 5150 images with class 3 damage, and 801 images with class 4 damage (fig. 2).

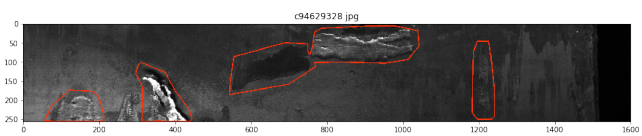

Fig. 1. Sample image from dataset

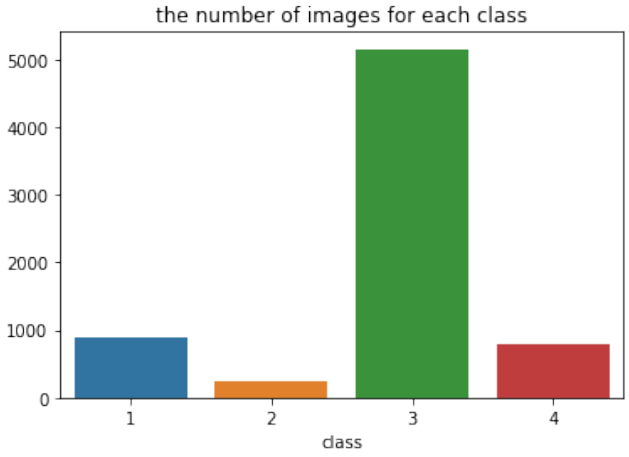

Fig. 2. Distribution diagram of images by classes

The training sample also contains 5,902 images of intact surfaces, 6239 images with 1 class damage, 425 images with 2 classes damage, and 2 images with 3 classes damage.

C. Networks architectures

This paper compares U-Net and DeepLabV3 architectures based on ResNet18, ResNet34, ResNet50, and E-Net architecture.

U-Net [15] is a Convolutional Neural Network (CNN) that was developed for the segmentation of biomedical images. The architecture looks like ”U” (fig. 3). This architecture has three parts: contraction, bottleneck, and expansion. The compression section is made up of compression blocks. Each block takes input and applies convolutional layers. After each block, the number of feature maps doubles. The lowest layer mediates between the compression layer and the expansion layer.

This architecture is based on an extension section. Like the compression section, it also consists of several expansion

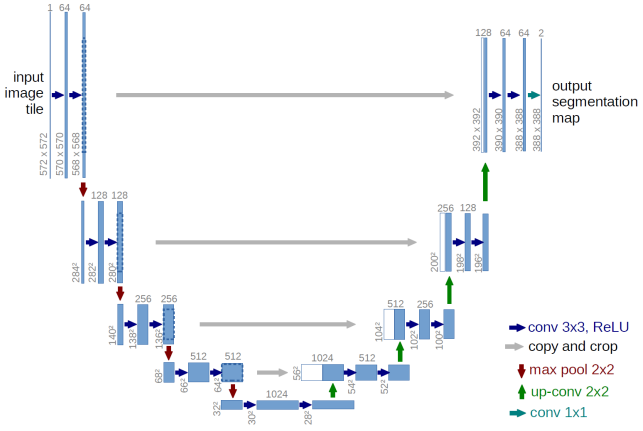

Fig. 3. U-Net architecture [**?**]

blocks. After each block, the number of feature maps used by the convolutional layer is halved to maintain symmetry. However, each time the feature maps of the corresponding compression layer are added to the input data. This ensures that the features of the image are not lost during reconstruction. The number of expansion units is the same as the number of compression units. After that, the image goes through another CNN layer with the number of feature maps equal to the number of the desired classes.

ResNet [16] is used in U-Net as an encoder, it is a powerful deep neural network with good performance [17]. The ResNet architecture allows you to build a very deep network, used skip connection (or shortcut connection) to link the input from the previous layer to the next layer without any modification of the input.

The second architecture for comparison was E-Net (Efficient Neural Network) [18], which allows real-time semantic pixel segmentation. The E-Net architecture (Fig. 4) is divided into several stages, which are highlighted by horizontal lines in the table. The sizes of the output data are given for the input image resolution 512×512.

Stage 1 consists of 5 bottleneck blocks. Stage 2 and 3 have the same structure, with the exception that stage 3 does not downsample the input at the beginning. These three first stages are the encoder. Stage 4 and 5 belong to the decoder. Fullconv is a bare deconvolution.

Despite the greater depth of E-Net, it contains 10 times fewer parameters than U-Net, which is good for the speed

Name	Type	Output size
initial		$16 \times 256 \times 256$
bottleneck1.0	downsampling	$64 \times 128 \times 128$
$4\times$ bottleneck1.x		$64 \times 128 \times 128$
bottleneck2.0	downsampling	$128 \times 64 \times 64$
bottleneck2.1		$128 \times 64 \times 64$
bottleneck2.2	dilated 2	$128 \times 64 \times 64$
bottleneck2.3	asymmetric 5	$128 \times 64 \times 64$
bottleneck2.4	dilated 4	$128 \times 64 \times 64$
bottleneck2.5		$128 \times 64 \times 64$
bottleneck2.6	dilated 8	$128 \times 64 \times 64$
bottleneck2.7	asymmetric 5	$128 \times 64 \times 64$
bottleneck2.8	dilated 16	$128 \times 64 \times 64$
Repeat section 2, without bottleneck2.0		
bottleneck4.0	upsampling	$64 \times 128 \times 128$
bottleneck4.1		$64 \times 128 \times 128$
bottleneck4.2		$64 \times 128 \times 128$
bottleneck5.0	upsampling	$16 \times 256 \times 256$
bottleneck5.1		$16 \times 256 \times 256$
fullconv		$C \times 512 \times 512$

Fig. 4. E-Net architecture. Output sizes are given for an example input of 512×512

of work and makes this model applicable for segmentation on mobile devices in real time.

The third architecture for comparison was DeepLabV3 [19]. The main feature of this architecture is the use of atrous convolution, which allows you to capture more context without increasing the number of parameters. This feature is illustrated in the fig. 5

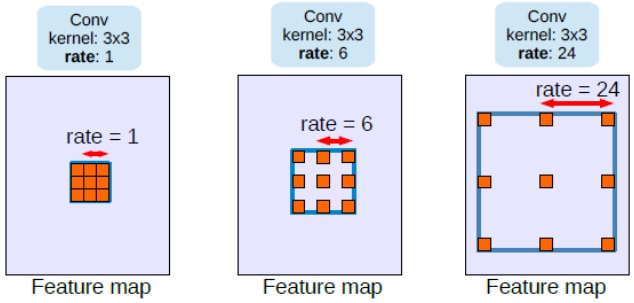

Fig. 5. DeepLab's atrous convolution

D. Training

The compared models were trained over 20 epochs. The transfer learning was used to train the encoder. As a base, we took ResNet trained on images of 1000 classes from the ImageNet [20] database. The Adam algorithm was chosen as the optimization method. Images of each class were divided into the training group (60%), validation group (20%), and the test group (20%) in a random way.

The models save with every improvement in the value of the loss function (binary cross-entropy). To combat class imbalance, the loss function is multiplied by weights that are inversely proportional to the proportion of the images of that class in the training sample. The following parameters are calculated for each model: loss, intersection over union (IoU)[21], Dice coefficient [22]. IoU and Dice metrics are calculated separately for images with defects and images without defects. The graph of the Dice coefficient obtained when training U-Net (ResNet50) is shown in the fig. 6.

III. RESULTS AND DISCUSSION

We have developed a python module for comparing models that provide the following functions:

- uniform data splitting;
- loading, transforming and saving data;
- mask visualization on images with defects;
- training, saving and download models;
- calculation of mean Dice and IoU metrics;
- calculation of accuracy, precision, and recall for each class;
- graphing of learning history.

A. Metrics

This paper uses two benchmarks or metrics to assess the quality of segmentation. The first is the use of Intersection

over Union (IoU) [21]. IoU is defined as follows:

$$IoU(X,Y) = \frac{X \cap Y}{X \cup Y}, \tag{1}$$

where X is the predicted set of pixels and Y is the actual pixel value. Dividing the intersection area by the union area gives the final result – intersection by union. The second criterion is evaluated by the Sørensen-Dice index (or average Dice similarity coefficient, DSC) [22]. The Dice coefficient is a binary measure of similarity that can be used to compare the pixel-by-pixel correspondence between a predicted segmentation and the corresponding underlying truth. The Dice coefficient is determined using the following equation:

$$DSC(X,Y) = \frac{2 * |X \cap Y|}{|X| + |Y|}, \tag{2}$$

where X is the predicted set of pixels and Y is the actual pixel value. The Dice coefficient is defined to be 1 when both X and Y are empty. Both methods take values from 0 to 1 to reflect the degree of intersection of the two regions.

The following metrics are also calculated for each test image as a whole:

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN}, \tag{3}$$

$$precision = \frac{TP}{TP + FP}, \tag{4}$$

$$recall = \frac{TP}{TP + FN}, \tag{5}$$

where TP - true positive, TN - true negative, FP - false positive, and FN - false negative predictions.

Picture prediction is considered correct for this class if the dice coefficient of intersection of the predicted mask and the true markup exceeds 0.2.

An example of a prediction and its corresponding metrics is shown in the figure 7.

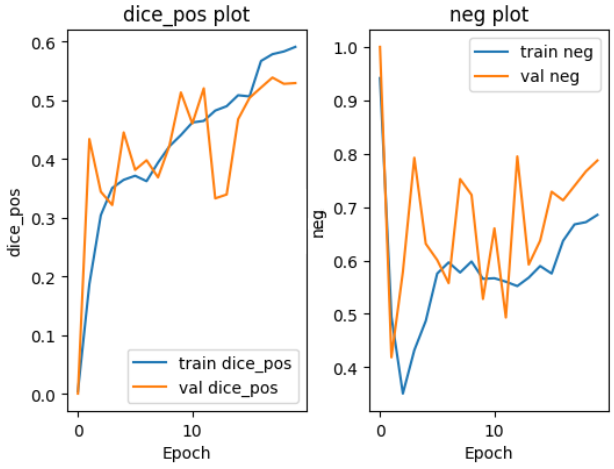

Fig. 6. Train Dice U-Net(ResNet50)

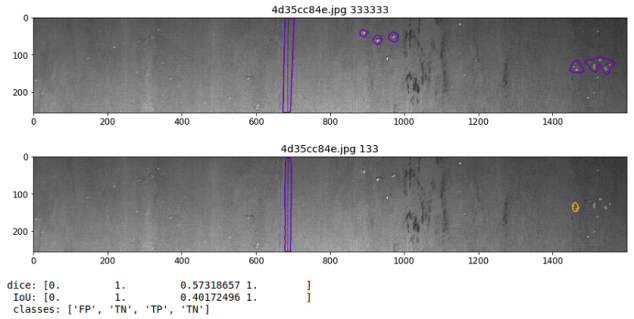

Fig. 7. An example of a tagged image (top) and prediction by the UNet model (ResNet50) (bottom) with the metrics values

B. Computing resources

To train the models, we used NVIDIA Jetson AGX Xavier [23] (technical specifications are shown in the table I). This microcomputer brings AI (Artificial Intelligence) capabilities to autonomous devices, providing up to 32 tera-operations per second. As part of the AI computing platform, it is equipped with the full suite of NVIDIA AI tools to quickly train and deploy neural networks. It supports applications developed with JetPack software development kits. The JetPack SDK is geared towards the tasks of self-driving machines and includes support for artificial intelligence, computer vision, multimedia, and more. As the world's first computer designed specifically for autonomous devices, the Jetson AGX Xavier is powerful enough for visual odometry, sensor fusion, localization and mapping, object recognition and routing, which are key to next-generation robots.

TABLE I
JETSON AGX XAVIER SPECIFICATION

GPU	512-core Volta GPU with Tensor Cores
CPU	8-core ARM 64-bit CPU, 8MB L2, 4MB L3
Memory	16 GB 256-Bit LPDDR4x — 137GB/s
Storage	32GB eMMC 5.1
DL Accelerator	(2x) NVDLA Engines
Vision Accelerator	7-way VLIW Vision Processor
Encoder/Decoder	(2x) 4Kp60 — HEVC/(2x) 4Kp60

C. Recognition quality and models speed

The training lasted 20 epochs. The main results of comparing the considered models in terms of segmentation quality and speed are shown in the table II.

TABLE II
COMPARISON OF MODELS

Model	img per sec	best val loss	val dice
E-Net	21.31	0.0197	0.577
U-Net(ResNet18)	7.93	0.0232	0.657
DeepLabV3(ResNet18)	4.91	0.0160	0.717
U-Net(ResNet34)	5.93	0.0246	0.574
DeepLabV3(ResNet34)	3.06	0.0151	0.728
U-Net(ResNet50)	4.76	0.0228	0.611

Graphs of changes in the value of the training loss function are shown in figure 8, and for validation in figure 9.

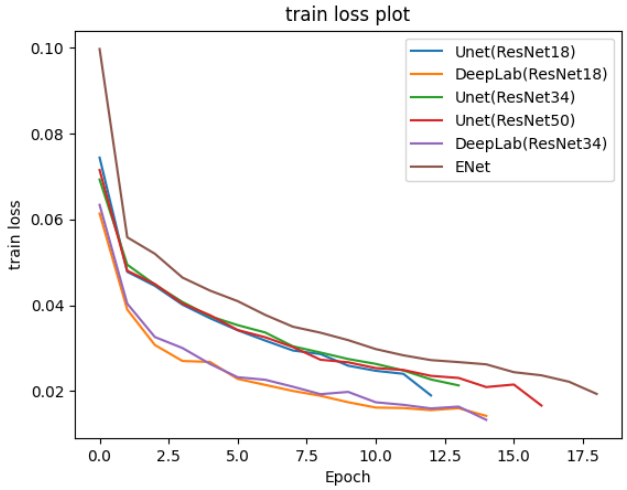

Fig. 8. Train loss

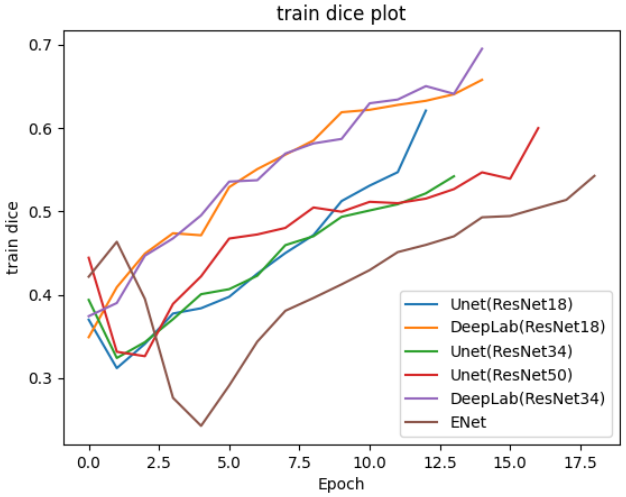

Fig. 10. Train Dice coefficient

Similar graphs of the Dice coefficient change are shown in 10, 11. The IoU graphs follow the shape of the graphs for the Dice coefficient, therefore, they are not shown.

The results of assessing accuracy, precision, and recall for each class are shown in the table III.

As you can see from table the E-Net architecture provides high speed gains, but lags behind in segmentation quality.

Among architectures based on ResNet 18, 34, 50, there is a regular improvement in segmentation accuracy with decreasing speed.

The DeepLabV3 architecture showed significant superiority in quality over the U-Net when using the same encoders.

Based on the analysis of the initial data and learning outcomes, it can be concluded that the training data is inaccurate marking, as evidenced by high detection rates for some classes, but low dice rates of the intersection coefficient of the predicted area with the real one.

The lack of second class data leads to the impossibility of identifying key features by the model, despite the correction of the weight of this class during training.

The insignificant difference between the quality indicators of different models and the lack of monotony in these indicators when comparing models on different classes also indicate insufficient data and inaccurate markup. The most indicative for comparing models quality are the metrics values for class 3, which is most widely represented in the training sample.

In the future, it is planned to consider the phased use of classifiers to improve the recognition accuracy and increase the system performance. At the first stage, a simple classifier is guaranteed to filter images without defects, which significantly reduces the load on subsequent deep classifiers that distribute defects into different classes. Also, for the implementation of the first stage, it is planned to consider the use of classical methods of computer vision without using machine learning.

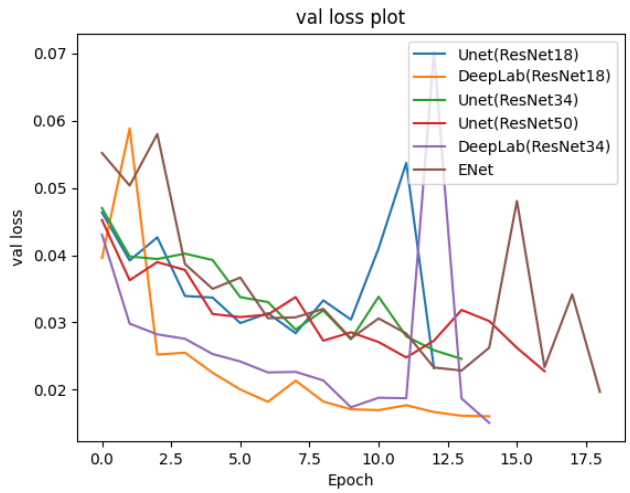

Fig. 9. Validation loss

Fig. 11. Validation Dice coefficient

TABLE III

PREDICTIONS QUALITY FOR EACH CLASS

class 1				
Model	acc	prec	recall	dice
E-Net	0.764	0.183	0.691	0.421
U-Net(ResNet18)	0.898	0.364	0.761	0.441
DeepLabV3(ResNet18)	0.920	0.464	0.909	0.460
U-Net(ResNet34)	0.609	0.141	0.898	0.239
DeepLabV3(ResNet34)	0.908	0.431	0.933	0.454
U-Net(ResNet50)	0.714	0.187	0.910	0.343
class 2				
Model	acc	prec	recall	dice
E-Net	0.885	0.145	1	0.557
U-Net(ResNet18)	0.884	0.125	0.872	0.437
DeepLabV3(ResNet18)	0.948	0.269	0.959	0.504
U-Net(ResNet34)	0.861	0.123	0.980	0.505
DeepLabV3(ResNet34)	0.959	0.320	0.980	0.524
U-Net(ResNet50)	0.879	0.137	0.960	0.525
class 3				
Model	acc	prec	recall	dice
E-Net	0.794	0.742	0.742	0.395
U-Net(ResNet18)	0.853	0.806	0.804	0.342
DeepLabV3(ResNet18)	0.902	0.913	0.825	0.394
U-Net(ResNet34)	0.814	0.793	0.712	0.379
DeepLabV3(ResNet34)	0.906	0.911	0.847	0.392
U-Net(ResNet50)	0.841	0.810	0.781	0.413
class 4				
Model	acc	prec	recall	dice
E-Net	0.874	0.329	0.962	0.537
U-Net(ResNet18)	0.950	0.564	0.924	0.491
DeepLabV3(ResNet18)	0.970	0.684	0.975	0.514
U-Net(ResNet34)	0.945	0.539	0.874	0.580
DeepLabV3(ResNet34)	0.970	0.686	0.968	0.525
U-Net(ResNet50)	0.946	0.541	0.918	0.537

IV. CONCLUSIONS

A method was developed for recognizing and classifying defects of metal surfaces by their images in real-time. For this purpose, a comparison of the E-Net, DeepLabV3 and U-Net models based on deep residual neural networks was carried out, their qualitative metrics were investigated on images of flat steel surfaces. The results showed that the proposed models can be used to detect surface defects.

The best segmentation accuracy was achieved using the DeepLabV3 architecture based on ResNet34. Simpler models showed worse generalizing properties.

At this stage for further use with the NVIDIA Jetson AGX Xavier microcomputer, the DeepLab architecture based on ResNet18 was chosen, since it is faster and the difference in accuracy with the ResNet34 is small.

A further direction of work is to divide the detection of defects into stages. The first stage is to filter out images without defects. The second stage is to carefully classify images with defects. This separation will improve the classification accuracy while reducing the computation time.

REFERENCES

[1] Kostenetskiy, P., Alkapov, R., Vetoshkin, N., Chulkevich, R., Napolskikh, I., & Poponin, O. (2019). Real-time system for automatic cold strip surface defect detection. FME Transactions, 47(4), 765-774.

[2] Luo, Q., Fang, X., Liu, L., Yang, C., & Sun, Y. (2020). Automated Visual Defect Detection for Flat Steel Surface: A Survey. IEEE Transactions on Instrumentation and Measurement, 69, 626-644.

[3] Wu, G. Online surface inspection technology of cold rolled strips //Multimedia, Kazuki Nishi, InTech. – 2010. – P. 205-232.

[4] Medina, R. Surface Defect Detection on Rolled Steel Strips by Gabor Fil-ters /R. Medina, F. Gayubo, L. González, D. Olmedo, J. Gómez, E. Zalama, J. Perán // VISAPP 2008 – International Conference on Computer Vision Theory and Appli-cations. – 2008. – p. 479-485

[5] Rail Surface Quality Nondestructive Examination Patent № RU2426069C1, IPC B21C51/00; G01B11/30, Kulakov S. Trofimov V. Applicants:G Obrazovatel Noe Uchrezhdenie Vysshego Professional Nogo Obrazovanija Sib G Ind Nyj Uni [Ru], Publication: 2011-08-10.

[6] State Registration Certificate for Computer Software Patent № 2013612245, Russian Federation. Defect Detector for Cold-Rolling Mill" / M.I. Kuzmin, N.A. Solovyov, D.A. Lesovoy. - № 2012661665; entry date December 27, 2012; date of registration in the Register of Computer Programs, February 19, 2013. - Published on 20.03.2013.

[7] Recognition of surface defects of cold-rolling sheets based on method of localities /Gergel V.P., Kuzmin M.I., Solovyov N.A.,Grishagin VA. //International Review of Automatic Control. - 2015. - Ko/.S, No 1. -P . 51 - 55.

[8] Konovalenko, I., Maruschak, P., Brezinová, J., Viňáš, J., & Brezina, J. (2020). Steel Surface Defect Classification Using Deep Residual Neural Network. Metals, 10(6), 846.

[9] Li, J and all. Real-time Detection of Steel Strip Surface Defects Based on Improved YOLO Detection Network // IFAC-PapersOnLine. 2018. Vol. 51(21). P. 76-81.

[10] Mohan, A.; Poobal, S. Crack detection using image processing: A critical review and analysis. Alex. Eng. J. 2018, 57, 787–798.

[11] Li, Y.; Li, G.; Jiang, M. An end-to-end steel strip surface defects recognition system based on convolutional neural networks. Steel Res. Int. 2017, 88, 60–68.

[12] Cui, W.; Zhang, Y.; Zhang, X.; Li, L.; Liou, F. Metal Additive Manufacturing Parts Inspection Using Convolutional Neural Network. Appl. Sci. 2020, 10, 545.

[13] Bernshteyn, M.L. (Ed.) Atlas Defects of Steel; Metallurgiya: Moscow, USSR, 1979; p. 188. (In Russian)

[14] Becker, D.; Bierwirth, J.; Brachthäuser, N.; Döpper, R.; Thülig, T. Zero-Defect-Strategy in the Cold RollingIndustry. Possibilities and Limitations of Defect Avoidance and Defect Detection in the Production of Cold-RolledSteel Strip; Fachvereinigung Kaltwalzwerke e.V., CIELFFA: Düsseldorf, Germany, 2019; p. 16.

[15] Ronneberger, O., Fischer, P., Brox, T., 2015. U-net: Convolutional networks for biomedical image segmentation, in: Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). Springer Verlag, pp. 234–241. doi:10.1007/978-3-319-24574-4-28

[16] M. Pak and S. Kim, "A review of deep learning in image recognition," 2017 4th International Conference on Computer Applications and Information Processing Technology (CAIPT), Kuta Bali, Indonesia, 2017, pp. 1-3, doi: 10.1109/CAIPT.2017.8320684.

[17] He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. arXiv 2015, arXiv:1512.03385v1.

[18] A. Paszke, A. Chaurasia, S. Kim, and E. Culurciello. E-net: A deep neural network architecture for real-time semantic segmentation. arXiv preprint arXiv:1606.02147, 2016.

[19] Chen L-C, Papandreou G, Schroff F, Adam H. Rethinking Atrous Convolution for Semantic Image Segmentation. arXiv e-prints arXiv:1706.05587, 2017.

[20] Frid-Adar M., Ben-Cohen A., Amer R., Greenspan H. (2018) Improving the Segmentation of Anatomical Structures in Chest Radiographs Using U-Net with an ImageNet Pre-trained Encoder. In: Stoyanov D. et al. (eds) Image Analysis for Moving Organ, Breast, and Thoracic Images. RAMBO 2018, BIA 2018, TIA 2018. Lecture Notes in Computer Science, vol 11040. Springer, Cham.

[21] M. A. Rahman and Y. Wang, "Optimizing intersection-over-union in deep neural networks for image segmentation," in Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 2016, vol. 10072 LNCS, pp. 234–244.

[22] Sörensen T. A method of establishing groups of equal amplitude in plant sociology based on similarity of species content // Kongelige Danske Videnskabernes Selskab. Biol. krifter. Bd V. № 4. 1948. P. 1-34.

[23] Nvidia Corp., "Jetson AGX Xavier Developer Kit — NVIDIA Developer," Developer.Nvidia.Com, 2018.