Artificial Intelligence in Industry 4.0: The future that comes true: AI

Karabegović, Isak; editor

2024-09-17

https://bastina.anubih.ba/handle/123456789/791

DOI: 10.5644/PI2024.215.03

Machine Learning Within Industry 4.0: From Decision Trees to Visual Transformer Architecture

Zoran Miljković[*1], Aleksandar Jokić[1], Milica Petrović[1]

Abstract: *Machine learning models have been revolutionizing the manufacturing industry for a decade now. Many powerful models have been developed to solve previously thought not solvable problems. In the era of rapid development of the machine learning field, the selection of optimal models for the considered problem within the Industry 4.0 setting is still a hard question. In the frist part of this work , we provide an overview of the use cases of machine learning for Industry 4.0. Afterwards, we provide details of the most commonly utilized machine learning models. Finally, the experimental evaluation is performed to compare the analyzed models for two different use cases, namely visual inspection and predictive maintenance. Two machine learning models are compared for each task in order to highlight the main differences and experimental results.*

Keywords: *Machine learning, Industry 4.0, visual transformer, artificial neural networks, convolutional neural networks,decision trees.*

1. Introduction

Artificial Intelligence (AI) is a crucial driver in the fourth industrial revolution, offering substantial advantages to the industrial sector through improved operational efficiency, cost reduction, and data-driven decision-making. Advanced machine learning algorithms based on AI excel in data analysis and process optimization[1], outperforming human-created algorithms in enhancing manufacturing productivity and efficiency[2]. These algorithms enable machines to operate continuously and undertake tasks that are either too laborious or dull for human workers. Moreover, AI-based algorithms can perform complex tasks that traditional algorithms struggle with, such as recognizing patterns in large datasets, making predictions from unstructured data, and adapting to new information without explicit programming. Furthermore, by providing comprehensive insights and predictive analytics, AI algorithms aid in more informed decision-making at all levels of an organization[3]. They can be used to identify trends, forecast outcomes, and provide recommendations, enabling management to make strategic decisions based on data.Another significant benefit of the implementation of AI regards the integration of various systems

[*1]University of Belgrade – Faculty of Mechanical Engineering Production engineering department
E-mail: zmiljkovic@mas.bg.ac.rs, ajokic@mas.bg.ac.rs, mmpetrovic@mas.bg.ac.rs

and processes within the industrial setup, ensuring seamless communication and interaction across different platforms and machines, leading to more cooperative and coordinated operations. Finally, these integrated processesrespond more quickly to market changes, and optimize their operations in ways that ensures competitive advantage, sustainability, and waste reduction[4].

Particularly, one of the production subsystems that can benefit extensively from AI is predictive maintenance. Machine learning-enabled predictive maintenance allows companies to anticipate equipment failures before they occur. This predictive capability helps reduce downtime and maintenance costs, as maintenance can be scheduled before a machine breaks down, preventing disruptive and expensive emergency repairs.

Vibration, acoustic, electrical, and temperature data from machinery can beused to detect anomalies or patterns that indicate impending failures[5]. Machine learning models such as Support Vector Machines [6] or Artificial Neural Networks [7] can classify vibration signals into regular or potential failure states, allowing for timely maintenance actions before a breakdown occurs. Machine learning algorithms, such as Random Forests [8] or Gradient Boosting [9], can integrate various data points (temperature, vibration, and operational hours) to predict when a piece of equipment might fail, which enables proactive maintenance scheduling.Learning algorithms can analyze sounds from equipment to detect changes that may indicate mechanical issues.

Deep learning models, particularly Convolutional Neural Networks (CNNs), are adept at processing audio data and identifying subtle changes in acoustic patterns [10] that might signify potential failures. Transformer architecture based on large language models can be utilized to use maintenance logs made by operators to predict future failures[11]. Data-driven digital twins[12] (virtual replicas of physical systems) can simulate and analyze the behavior of machinery under different conditions. By utilizing AI algorithms within these simulations, companies can predict failures and understand potential maintenance needs without interrupting actual operations. Deep learning models, particularly Long Short-Term Memory networks[13] are effective in analyzing time-series data from sensors to detect patterns that precede equipment failures. These models can learn from historical data and make accurate predictions about the future machinery state.Having that in mind, in this chapter, the analysis of decision trees and artificial neural networks for machine failure prediction will be made, based on the predictive maintenance dataset[14].

Moreover, AI-basedmodelscan be used to improve quality control processes in manufacturing. By analyzing images or sensor data, these models can identify product defects or anomalies, ensuring that only items meeting the highest quality standards reach customers. Particularly, traditional inspection processes are being turned into more efficient, accurate, and automated

operations. CNNs are particularly effective for image processing tasks, enabling the identification of various defect types, such as cracks, misalignments, or incorrect dimensions, with high accuracy. Deep learning models can be trained to recognize and classify different types of surface flaws on materials or parts. For example, a neural network can differentiate between acceptable variations and defects on a product surface. Unsupervised learning algorithms, e.g., Autoencoders[15], are used to learn the normal patterns of data and detect deviations that signify defects or quality deviations. Finally, Large Language Models (LLMs) can analyze customer feedback, reviews, or reports to identify common quality issues or trends[16]. This information can be used to improve customer feedback and, therefore product design and manufacturing processes. In some inspection applications, CNN models are integrated with robotic systems to automate the inspection process. Robots equipped with real-time vision systems [17]can conduct consistent and detailed inspections and relieve human workers from dull tasks[18]or environmentally hazardous surroundings. Therefore, in this chapter, the utilization of CNNs for the visual inspection of casting parts will be performed and analyzed.

Another aspect of Industry 4.0 is the integration of machine learning algorithms and robotics, which transforms manufacturing environments and creates smarter, more flexible, and more highly efficient processes[19]. This integration allows robots to perform tasks with greater autonomy, adaptability, and precision. Autonomous mobile robots enhanced with AI-based control algorithms[20] navigate complex industrial environments independently, avoiding obstacles, optimizing routes[21], and transporting materials[22]. They use sensor data processed by machine learning algorithms to interpret their surroundings and make real-time decisions. Moreover, AI-driven vision systems enable robots to interpret visual data [23]and carry out complex tasks like quality control, inspection, and object recognition. These systems use advanced machine learning-based image processing algorithms, such as CNNs, to analyze images or video streams in realtime.

Furthermore, LLMs can be used toenhance the interaction between humans and robots, enabling more intuitive and effective collaboration. Natural language processing (NLP) and machine learning based on LLMs allow robots to understand and respond to human commands, while predictive algorithms help them anticipate human actions and adjust robot behavior accordingly. Collaborative robots can work alongside humans without the need for physical barriers. All these capabilities are enabled by machine learning models, all while ensuring safety and efficiency. Machine learning models help collaborative robots to improve their performance over time, learning from each human interaction.

In summary, artificial intelligence, especially machine learning, represents one of thecornerstones of Industry 4.0, driving significant

transformations across industries by enabling smarter, more efficient, and more adaptive manufacturing systems.

2. Machine Learning Models

A machine learning model is a computational framework of a system or process whose parameters are learned by utilizing data and a specific training algorithm. After the training, the model can identify patterns or relationships within the data, allowing it to make a prediction without being explicitly programmed for a specific task. More formally, the machine learning model (1) is a function f with the set of tunable parameters θ that can map the input data X to the output prediction Y. The input/output pairs of data define the dataset (D). The training process consists of the optimization of parameters θ based on the defined loss function.

$$Y = f\left(X;\theta\right) \tag{1}$$

A well-trained machine learning model needs to generalize well to new and unseen data that are usually part of the validation and test datasets. Generalization is achieved through techniques such as cross-validation and regularization.The following sections will provide the definition of machine learning models such as decision trees, artificial neural networks, and convolution neural networks.

2.1. Decision trees

Decision trees are one of the most commonly utilized machine learning algorithms for both classification and regression tasks(Fig. 1). They work by incrementally developing a decision tree by breaking down a dataset (feature space) into smaller subsets based on conditions regarding the input features. The final result is a tree with root node, decision nodes, and leaf nodes that can be used to make predictions based on the features of a new data point.

Figure 1. Abstract illustration of a decision tree

The root node represents the entire dataset (D), which is then split into two or more sets based on the most dominant threshold of feature. The goal is to find the combination of featuresand thresholds (θ_t) that best separates the dataset at node t (2) and (3).

$$D_t^{left} = \left\{ \left(X, Y \right) \in D \middle| X_i \le \theta_t \right\}, \tag{2}$$

$$D_t^{right} = \left\{ \left(X, Y \right) \in D \middle| X_i \le \theta_t \right\}, \tag{3}$$

To determine the best split, metrics such as Gini impurity (4) or entropy (5) are used:

$$G\left(t \right) = 1 - \sum_{k=1}^{K} p_k^2 , \tag{4}$$

$$H\left(t \right) = -\sum_{k=1}^{K} p_k \log p_k , \tag{5}$$

where K is the number of classes, and p_k represents the probability that the sample belongs to the k-th class.Entropy measures the randomness or uncertainty in the dataset. Gini impurity measures the probability of a randomly chosen element being incorrectly classified according to the distribution of labels in the subset.Different algorithms are used to determine how to make a split and, therefore, create a decision tree Classification and Regression Trees[24],

Iterative Dichotomiser 3[25], and C4.5[26]. These algorithms partition the data into subsets that have the highest purity or lowest impurity.

$$\theta_t^* = \arg\min_{X_i,\theta_t}\left(\frac{n_t^{left}}{n_t}\cdot G\left(D_t^{left}\right)+\frac{n_t^{right}}{n_t}\cdot G\left(D_t^{right}\right)\right) \tag{6}$$

where θ_t^* representsthe optimal threshold value for the optimal feature $X_i$at node t, and n represents the number of samples. Decision nodesare utilized to split each subset of the feature space further. These nodes can be split further into additional decision nodes or become a final (leaf) node. Leaf nodes represent the outputs of the decision tree where no further splitting is possible. In classification, it represents the class of the data points in this node. In regression, it typically represents the mean of the target values. Once the tree is built, it can be used to make predictions. For a new data point, the decision tree is used by testing the relevant conditions (decision and root nodes) until a leaf node is reached.

2.2. Artificial Neural Networks

Artificial Neural Networks (ANNs) represent machine learning models that are inspired by human neural networks. They are utilized for approximation of complex functions, classification, and time series analysis[27]. The structure of the ANN consists of layers of neurons that are all interconnected with tunable weights. The ANN has a minimum of three layers, the input layer, where the input vectors are passed; the hidden layers, which represent the hyperparameter of the model; and the output layer used to generate output or result of a network. The number of neurons in the input and output layer is defined by the number of features in input and output vectors, while the number of neurons in each hidden layer is also a hyperparameter. The learning process of an ANN is performed by adjusting the weight parameters until the difference (defined by a loss function) between desired and predicted output vectors is minimal.

The input vector is defined as $\mathbf{X} = (x_1, x_2, \ldots, x_n)$, where n represents the number of features. The output of all neurons in the hidden and output layers are defined by (7)

$$h^l = f\left(\boldsymbol{W}^l h^{l-1} + b^l\right) \tag{7}$$

where h represents the output of each neuron in layer l, f is the activation function, $\boldsymbol{W}$ is a matrix of weights between l-1 and l layer, and b represents the bias value. The type of activation function used is another hyperparameter of the ANN; some of the most utilized are:

- Sigmoid activation function $f_s(x) = \left(1 + e^{-x}\right)^{-1}$
- Hyperbolic tangent activation function $f_t(x) = \dfrac{e^{x} - e^{-x}}{e^{x} + e^{-x}}$
- Rectified Linear unit (ReLU) activation function $f_r(x) = \max(0, x)$
- Gaussian error Linear Unit (GeLU) activation function $f_g(x) = \dfrac{x}{2}\left(1 + \tanh\left(\sqrt{\dfrac{2}{\pi}}\left(x + 0.445x^3\right)\right)\right)$

The formal definition of the general training algorithm is given in equation (8):

$$\boldsymbol{W}^{new} = \boldsymbol{W}^{old} + \eta \frac{\partial L}{\partial \boldsymbol{W}}, \tag{8}$$

where η represents the learning rate, and L is the loss function.

2.3. Convolutional Neural Networks

Convolutional Neural Networks (CNNs) representa group of artificial neural network architectures that utilize convolution as a primary processing operation.Unlike the standard ANNs defined in Section 2.2, CNNs do not have all the neurons connected in subsequent layers, and the architecture is generated by utilizing different layers. Another distinguishing factor is that CNNs are typically utilized in the Deep Learning (DL) domain, while ANNs are used in traditional Machine Learning (ML). Even though DL is a subdomain of ML, the main difference is that DL is used when there are large and complex datasets, while for ML,some sort of feature engineering is required first. Moreover, DL models are characterized by utilizing an enormous number of parameters and many subsequent layers.

One of the most commonly used layers in CNNsisthe Convolutional Layer, where the main parameters are the filter size, stride, padding, and number of filters. After the convolutional layer, activation layers are usually implemented with a ReLU activation function. To minimize the resolution of the input feature maps and, therefore parameters in the network, the average or max pooling layers are utilized. Regularization is achieved by using dropout or batch norm layers. A detailed explanation of the individual layers used in CNNs can be found in [28]. Numerous CNNs have been proposed throughout the years for different problems, mainly in the computer vision domain. Some of the most commonly utilized architectures include alexnet[29], VGG[30], resnet[31], mobilenet[32], and densenet[33].

2.4. Visual Transformer Architecture

A Visual Transformer (ViT)[34] is a type of artificial neural network architecture developed for computer vision tasks, leveraging the power of transformerarchitectures that have shown remarkable success in natural language processing. The entire ViT architecture is given in Fig. 2.

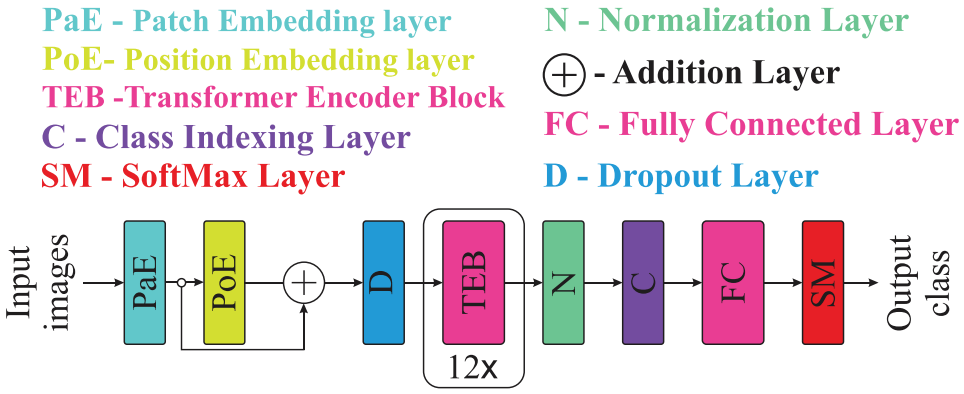

Figure 2. Visual Transformer (ViT) architecture.

As can be seen, the input images are processed through a sequence of layers, with the main block of layers defined within the Transformer Encoder Block (TEB), which is repeated 12 times. The output of ViT architecture is a vector that represents the class of the input image.

The ViT utilizes the process of dividing input images into non-overlapping patches of fixed size. Each patch (shown in green – Fig. 3) is reshaped into a vector, and a fully connected layer is applied to each vector to map it to anembedding dimension by using the Patch Embedding layer (PaE). This process is shown in Fig. 3. Each patch embedding layer output value is represented with a green rectangular cuboid. The concatenation layer is utilized to add an additional vector of weights at thestart of the output (shown with a black rectangular cuboid), which is later used to make a prediction of the class output.

Afterwards, the whole output (all green and black vectors) is multiplied with learnable weights from the PositionEmbedding layer (PoE)and added with the values before multiplication by using skip connection and addition layer (Fig. 2.).

Unlike Recurrent Neural Networks (RNNs) or CNNs, which have an inherent notion of sequence and locality, transformers process the entire image simultaneously.

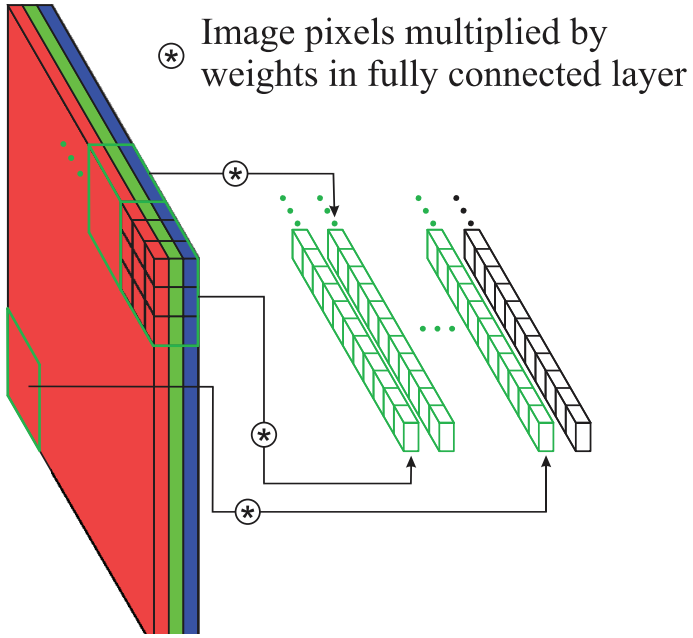

Figure 3. Patch Embedding layer (PaE)layer of ViT architecture

Therefore, there is a lack of a built-in mechanism to understand the sequential order of the input image patches, and the position embedding layer plays a crucial role in enabling the model to understand the order of image patches (and pixels) in the input sequence. The rest of the ViT architecture consists of several blocks of transformer encoder structure, which is shown in Fig. 4.

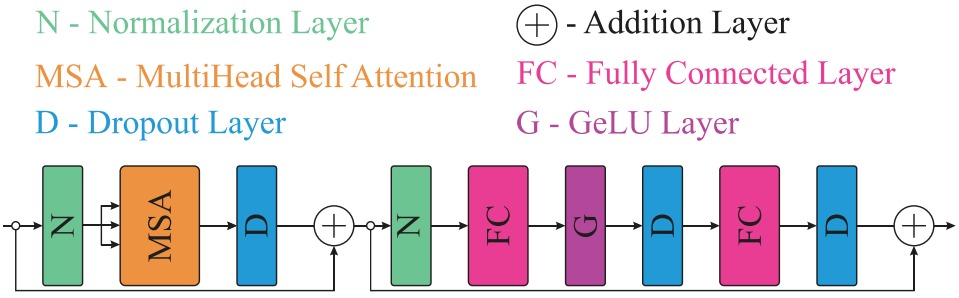

Figure 4. Transformer Encoder Block (TEB) structure

The Transformer Encoder Block (TEB) consists of the two parts separated by different skip connections and addition layer. The first part includes the Normalization layer that performschannel-wise normalization, the MutliHead Self Attention block, which will be covered in detail later, and dropout layers that are used for regularization purposes. In the second part, after the

normalization layer, the fully connected layer (explained in Section 2.2) is activated with the GeLU activation function. Afterwards, the dropout layer, fully connected layer, and another dropout layer are connected, and their final output is added to the input of the second part of TEB. Finally, the core of the transformer architecture is a MultiHead Self Attention (MSA) block (Fig. 5). Within MSA, the input values are copied three times, and they represent Query, Key, and Value matrices. These matrices are connected to three different fully connected layers, and the output of the key and query is multiplied with the matrix multiplication layer.

Afterwards, the output matrix is divided with the square root of the dimensions of the matrix within the scale layer, and the probabilities are generated with SoftMax. The output of those layers represents the attention filter that maps the importance between image patches. When the value output is multiplied with the attention filter, the result represents the filtered value matrix (attention map) that focuses on individual objects within the image. Twelve of these Self-attention heads (all with different weight values, focusing on other objects/parts of the image) are concatenated within the concatenation layer and connected with another fully connected layer to provide the final output of MSA.

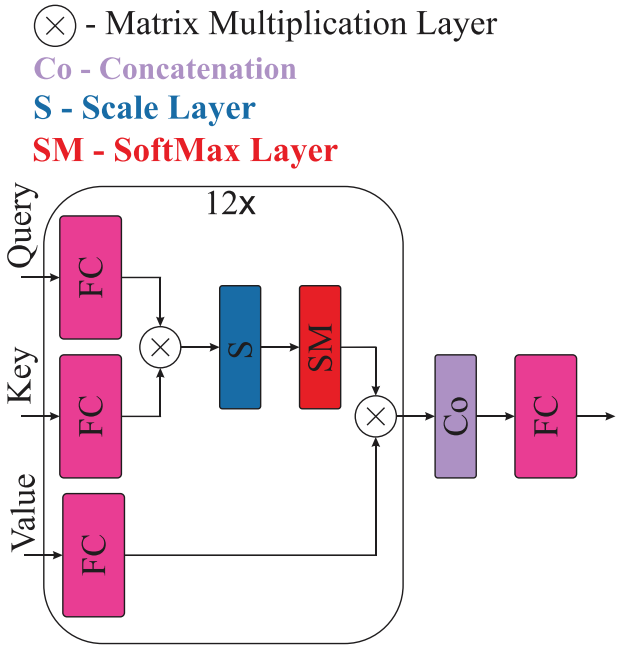

Figure 5. MultiHead Self Attention (MSA) block structure

3. Machine Learning Models within Industry 4.0 Settings: Experimental Results

The experimental evaluation is performed on the publicly available datasets [35][14]. The first dataset incorporates synthetic data regarding the predictive maintenance of milling machine based on seven features.The dataset contains 10000 data points;a few data samples are shown in Table 1. *Type* represents the product quality requirements (H – high, M – medium, L – low), while *Target* represents the output classes, 1 –failure (machine), 0 – non-failure. Even though the dataset has seven features, only six have been used for machine learning since product ID is just a randomly assigned number to each part and provides no additional information regarding the machine itself. The machine learning models are used to predict when the analyzed milling machine will fail in order to plan ahead for the maintenance process.The distributions of individual features within the dataset are given in Fig. 6.

Table 1. Samples of the predictive maintenance dataset

No.	Product ID	Type	Air temperature [K]	Process temperature [K]	Rotational speed [RPM]	Torque [Nm]	Tool wear [min]	Target
1	29462	H	298.8	309.2	1425	53.9	135	0
2	14909	M	298.9	309.2	1412	44.1	140	0
3	47230	L	298.9	309.1	2861	4.6	143	1
998	57153	L	298.5	308.2	1444	40.5	170	0
999	57154	L	298.6	308.2	1361	68.2	172	1
1000	24835	M	298.6	308.3	1589	34.1	174	0

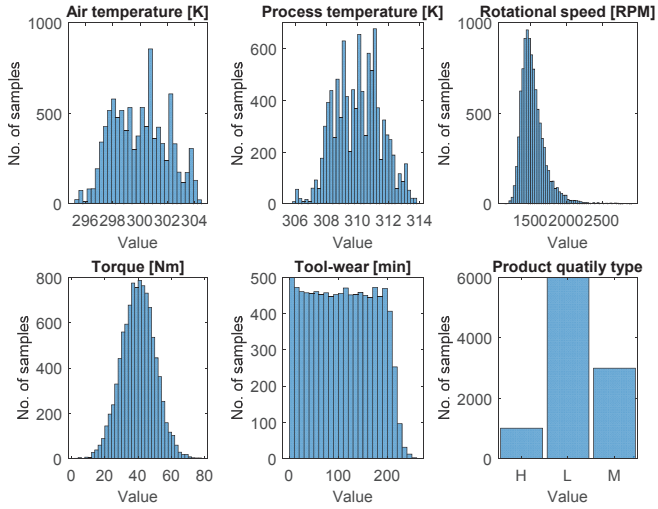

Figure 6. Distributions of features from the dataset.

As it can be seen, there is a wide variety in feature space with normal, uniform, and skewed distributions. After the data analysis is performed, machine learning can begin. As the input vector is relatively small (having six values), the utilization of CNNs or transformer architecture would probably result in overfitting.

Therefore, the problem of predictive maintenance is performed by utilizing artificial neural networks (fully connected networks) and decision trees. All the algorithms have been implemented using the Deep learning toolbox within MATLAB R2023b. The architecture of the utilized ANN is shown in Fig. 7. As it can be seen, the architecture of ANN is 6 $[50\text{-}30\text{-}10]_3$ 1. We utilized three hidden layers with 50, 30, and 10 neurons receptivity, whereas the first hidden layer has ReLU activation function, and two other hidden layers use tanh. Since the problem is binary classification, the output layer utilizes a standard sigmoid activation function.

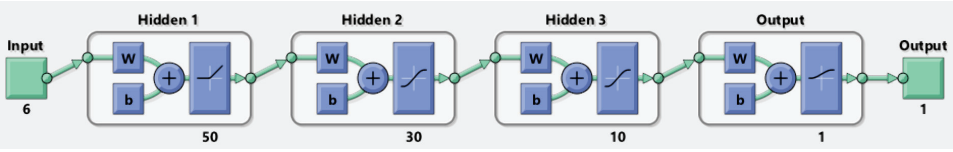

Figure 7. Architecture of the ANN

The training process of the ANN is shown in Fig. 8. The utilized training algorithm is Scaled conjugate gradient descent [36]with cross-entropy loss function. The dataset is randomly divided into training (80%), validation (10%), and test (10%) subsets.

The other machine learning model utilized for the problem of predictive maintenance is decision trees, also implemented within MATLAB Deep learning toolbox. The data fed into the model is the same as in the case of ANN. The utilized splitting criterion is Gini impurity; the maximum number of splits is set to 100 with no surrogates. The K-fold cross-validation is performed to ensure the generalization properties of the model. After the training is completed, the output tree with classes for each leaf is given in Fig. 9. The leaf nodes are represented with blue dots, while the decision nodes are shown with blue triangles.

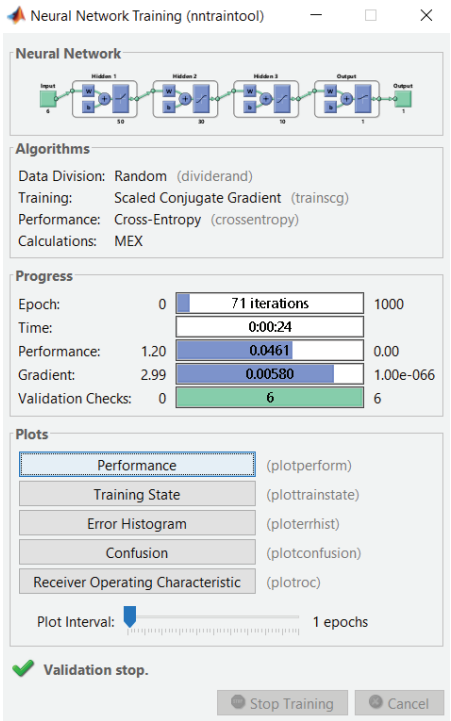

Figure 8. Training process of the ANN

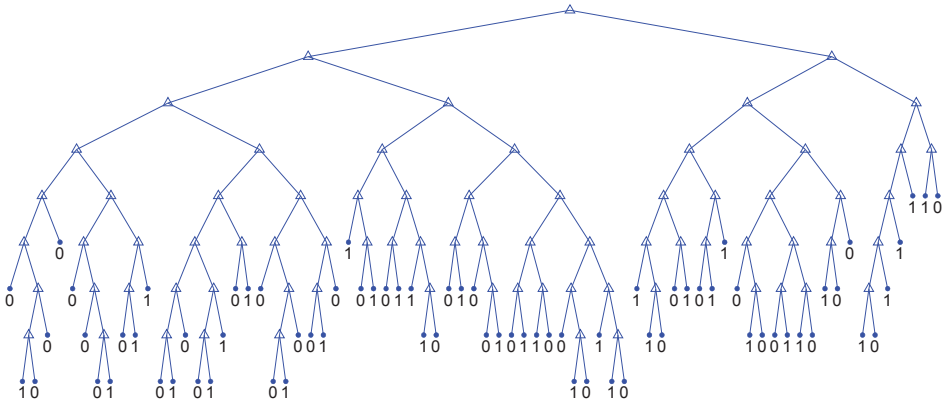

Figure 9. Decision tree

Moreover, as the entire decision tree would be hard to visualize, we present a part of the decision tree with decision values, decision node conditions and values in Fig. 10. As can be seen, the primary decision is based on the value of torque with a splitting value of 65.

The results are compared after the ANN and decision tree training are completed. The results are shown with two confusion matrices in Fig. 11.

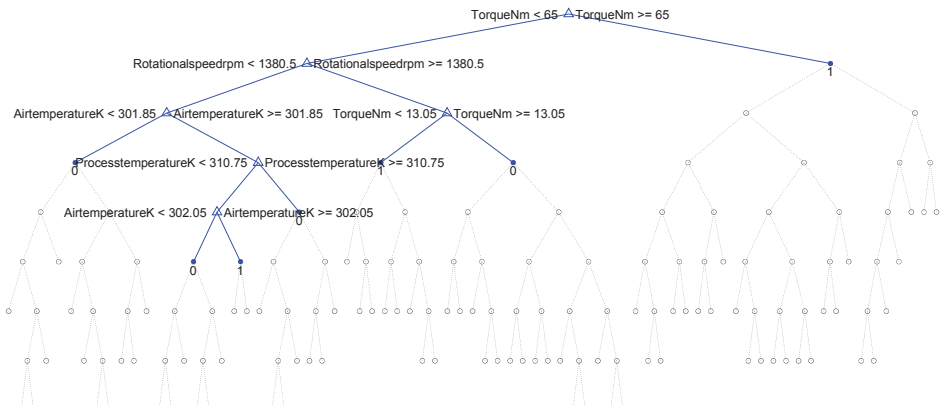

Figure 10. Decision tree with decision node conditions.

When the main parts of the confusion matrices are compared, the results seem similar. The results of the decision tree algorithm do not add to 100% due to the rounding of numbers to the first decimal point. Both ML models achieve 98.2% accuracy in detecting that non-failure will occur. Decision trees predict 0.01% more occurrences of failure compared to ANN. However, the decision trees also predict more non-failures when there is one, which means ANNs do better on this metric. On the other hand, ANNs also predict 0.01% more failure when there is none, which is a better occurrence compared to the last one. The experimental evaluation suggests that decision trees for the considered problem achieve better overall results.Another benefit of using a decision tree compared to ANN is the interpretability of the model. By using decision trees, it is always clear why some input generated predicted class based on the tree and the value of the decision variables. However, for ANN, it is not so easy to interpret why some weights in the network have considered values, and it is much harder to interpret why some input generated its paired output. Therefore, in the real-world Industry 4.0 setting, it is significantly easier to implement decision trees due to their ease of interpretation for humans.

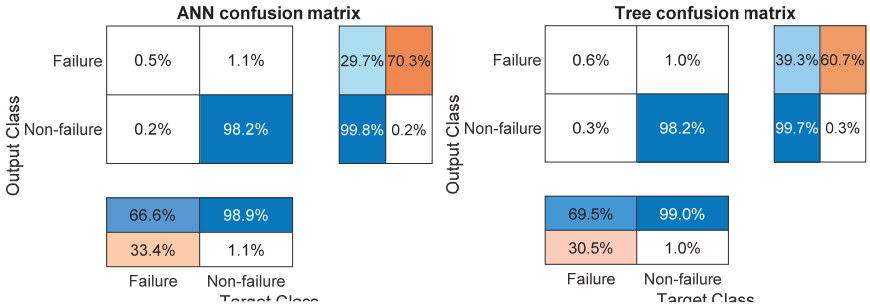

Figure 11. Confusion matrices of the analyzed models

The second experimental evaluation is performed on the visual inspection task. The considered dataset contains images for visual inspection of casting parts. Images are classified into two categories ok parts and parts with defects. The considered images are grayscale with aresolution of 512×512. The dataset is divided into trainingand testing subsets (130 images), whereas we further divide the training dataset into validation (130 images) and training (1040 images). Three data augmentation techniques are considered: rotating the image ±90 degrees, random scaling of the images, and mirroring the image along the Y axis. The samples of the correctly cast parts are shown in the first row of Fig. 12, while parts with defects are given in the second row.

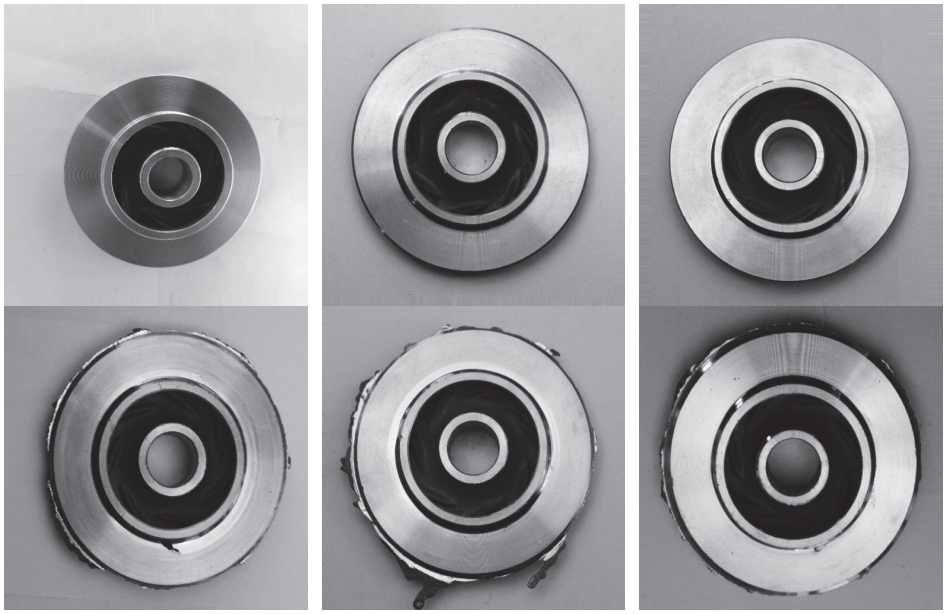

Figure 12. Sample images from the visual quality inspection dataset

A deep learning toolbox within MATLAB R2023b was utilized for the training of deep neural networks. For the considered problem, two deep neural networks are utilized:VisualTransformer (ViT) [34]and a densely connected convolutional network [33]. Both networks are pretrained on imagenet dataset and fine-tuned for the considered problem. The utilized dense network has an overall of 707 layers, with more than 800 connections between them. On the other hand, the utilized ViT has 143 layers with 167 connections. Therefore, since the size of networks differs significantly, to make a fair comparison, the ViT was trained for eight epochs, while the dense network was trained for four. Other training parameters were the same: mini-batch size of 12, Adam optimizer[37], and initial learning rate of 0.0001. The training process for dense network and ViT can be seen in Figs. 13 and 14, respectively.

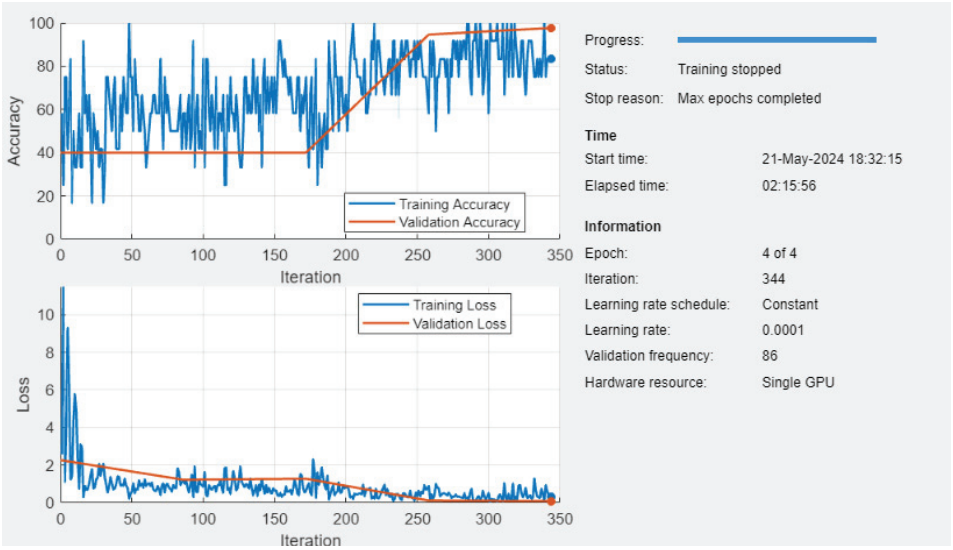

Figure 13. Learning process of the dense net

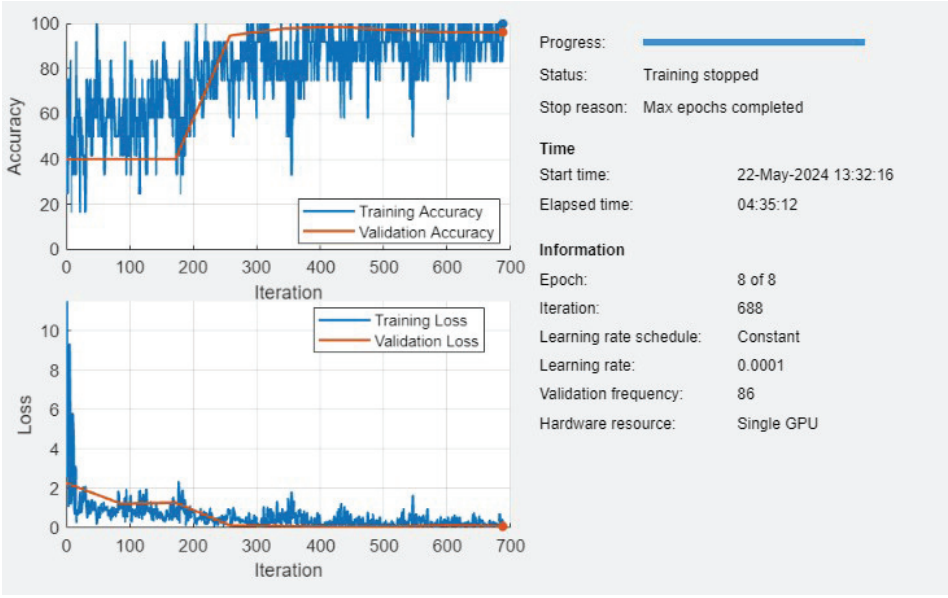

Figure 14. Learning process of ViT

As it can be seen, the loss function on both training and validation datasets gradually decreases during the training process for both networks. Also, training and validation accuracy increases and oscillates around 90%, indicating that the training process is progressing well. The final results of both networks are shown with test set confusion matrices in Fig. 15.

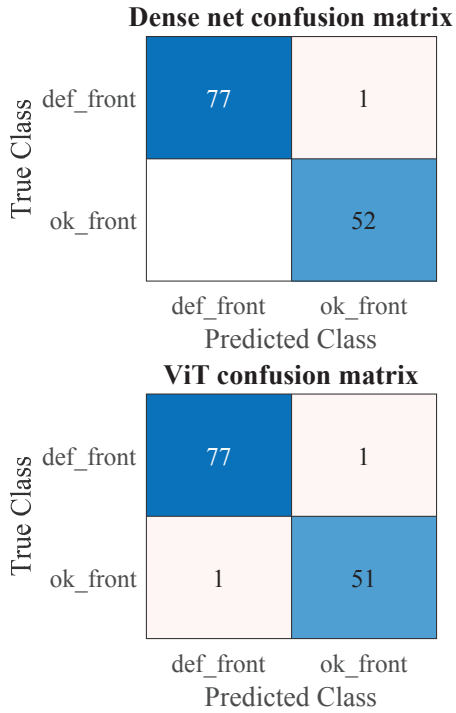

Figure 15. Confusion matrices of the dense net and ViTmodels

As shownin Fig. 15, both machine learning neural networks perform exceptionally well, achieving around 99% accuracy on the test set. The dense network performed slightly better by correctly classifying one sample more than the ViT transformer.

4. Conclusion

In this chapter, the authors analyzed the usage of different machine learning models within the Industry 4.0 domain. The main focus of the experimental evaluation was on the tasks of predictive maintenance and visual inspection. For predictive maintenance, machine learning models, namely, fully connected artificial neural networks, are compared to the decision trees on the publicly available dataset. The experimental evaluation shows a similar level of accuracy in detecting machine failure, with a slight advantage shown by decision trees. Another highlighted factor regarding the decision between these two machine learning models is the interpretability of the system, whereas decision trees have a significant advantage. The second experimental evaluation included the comparison of deep learning models on visual inspection task. Visual transformer architecture is compared to the dense convolution neural network. Both networks were pretrained on the ImageNet dataset and fine-tuned for visual inspection. The experimental results show high levels of accuracy (around 99% on the test set) of both analyzed networks, with a small advantage achieved by the densenet.

5. Acknowledgment

This research has been financially supported by the Ministry of Science, Technological Development, and Innovation of the Serbian Government, through the project "Integrated research in macro, micro, and nano mechanical engineering – Deep learning and cybersecurity of cyber-physical systems within Industry 4.0" (contract No. 451-03-65/2024-03/200105, 05/02/2024).

6. References

[1] Petrović, M., Mitić, M., Vuković, N., Miljković, Z.: Chaotic particle swarm optimization algorithm for flexible process planning, *The International Journal of Advanced Manufacturing Technology*, Vol. 85, No. 9–12, pp. 2535–2555, 2016.

[2] Miljković, Z., Petrović, M.: Application of modified multi-objective particle swarm optimisation algorithm for flexible process planning problem, *International Journal of Computer Integrated Manufacturing*, Vol. 30, No. 2–3, pp. 271–291, 2017.

[3] Jokić, A., Petrović, M., Miljković, Z.: Mobile robot decision-making system based on deep machine learning, in *9th International Conference on Electrical, Electronics and Computing Engineering (IcETRAN 2022)*, 2022, pp. 653–656.

[4] Petrović, M., Vuković, N., Mitić, M., Miljković, Z.: Integration of process planning and scheduling using chaotic particle swarm optimization algorithm, *Expert Systems with Applications*, Vol. 64, pp. 569–588, 2016.

[5] Vorkapić, N., Kokotović, B., Živanović, S.: Wavelet transform packet for signal analysis using standard deviation and signal spectrum, *Innovative Mechanical Engineering*, Vol. 3, No. 1, pp. 28–38, 2024.

[6] Goyal, D., Choudhary, A., Pabla, B. S., Dhami, S. S.: Support vector machines based non-contact fault diagnosis system for bearings, *Journal of Intelligent Manufacturing*, Vol. 31, pp. 1275–1289, 2020.

[7] Zhang, W., Peng, G., Li, C.: Bearings fault diagnosis based on convolutional neural networks with 2-D representation of vibration signals as input, in *MATEC web of conferences*, 2017, Vol. 95, p. 13001.

[8] Behera, S., Choubey, A., Kanani, C. S., Patel, Y. S., Misra, R., Sillitti, A.: Ensemble trees learning based improved predictive maintenance using IIoT for turbofan engines, in *Proceedings of the 34th ACM/SIGAPP symposium on applied computing*, 2019, pp. 842–850.

[9] Aziz, N., Akhir, E. A. P., Aziz, I. A., Jaafar, J., Hasan, M. H., Abas, A. N. C.: A study on gradient boosting algorithms for development of AI monitoring and prediction systems, in *2020 International Conference on Computational Intelligence (ICCI)*, 2020, pp. 11–16.

[10] Kumar, A., Gandhi, C. P., Zhou, Y., Kumar, R., Xiang, J.: Improved deep convolution neural network (CNN) for the identification of defects in the centrifugal pump using acoustic images, *Applied Acoustics*, Vol. 167, p. 107399, 2020.

[11] Usuga-Cadavid, J. P., Lamouri, S., Grabot, B., Fortin, A.: Using deep learning to value free-form text data for predictive maintenance, *International Journal of Production Research*, Vol. 60, No. 14, pp. 4548–4575, 2022.

[12] Rathore, M. M., Shah, S. A., Shukla, D., Bentafat, E., Bakiras, S.: The role of ai, machine learning, and big data in digital twinning: A systematic literature review, challenges, and opportunities, *IEEE Access*, Vol. 9, pp. 32030–32052, 2021.

[13] Jiang, Y., Dai, P., Fang, P., Zhong, R. Y., Zhao, X., Cao, X.: A2-LSTM for predictive maintenance of industrial equipment based on machine learning, *Computers & Industrial Engineering*, Vol. 172, p. 108560, 2022.

[14] Matzka, S.: Explainable artificial intelligence for predictive maintenance applications, in *2020 third international conference on artificial intelligence for industries (ai4i)*, 2020, pp. 69–74.

[15] Tang, T.-W., Kuo, W.-H., Lan, J.-H., Ding, C.-F., Hsu, H., Young, H.-T.: Anomaly detection neural network with dual auto-encoders GAN and its industrial inspection applications, *Sensors*, Vol. 20, No. 12, p. 3336, 2020.

[16] Shi, J., Li, J., Ma, Q., Yang, Z., Ma, H., Li, L.: CHOPS: CHat with custOmer Profile Systems for Customer Service with LLMs, *arXiv preprint arXiv:2404.01343*, 2024.

[17] Jokić, A., Petrović, M., Miljković, Z.: Semantic segmentation based stereo visual servoing of nonholonomic mobile robot in intelligent manufacturing environment, *Expert Systems with Applications*, Vol. 190, p. 116203, 2022.

[18] Miljković, Z., Mitić, M., Lazarević, M., Babić, B.: Neural network Reinforcement Learning for visual control of robot manipulators, *Expert Systems with Applications*, Vol. 40, No. 5, pp. 1721–1736, 2013.

[19] Petrović, M., Miljković, Z., Jokić, A.: A novel methodology for optimal single mobile robot scheduling using whale optimization algorithm, *Applied Soft Computing*, Vol. 81, p. 105520, 2019.

[20] Jokić, A., Petrović, M., Miljković, Z.: Methods for visual servoing of robotic systems: A state of the art survey (in Seriban), *Tehnika*, Vol. 73, No. 6, pp. 801–816, 2018.

[21] Vuković, N., Mitić, M., Miljković, Z.: Trajectory learning and reproduction for differential drive mobile robots based on GMM/HMM and dynamic time warping using learning from demonstration framework, *Engineering Applications of Artificial Intelligence*, Vol. 45, pp. 388–404, 2015.

[22] Petrović, M., Jokić, A., Miljković, Z., Kulesza, Z.: Multi-objective scheduling of a single mobile robot based on the grey wolf optimization algorithm, *Applied Soft Computing*, Vol. 131, p. 109784, 2022.

[23] Miljković, Z., Vuković, N., Mitić, M., Babić, B.: New hybrid vision-based control approach for automated guided vehicles, *The International Journal of Advanced Manufacturing Technology*, Vol. 66, No. 1–4, pp. 231–249, 2013.

[24] Breiman, L., Jerome, F., R.A., O., Stone, C. J.: *Classification and regression trees*. Chapman and Hall/CRC, 2017.

[25] Quinlan, J. R.: Induction of decision trees, *Machine learning*, pp. 81–106, 1986.

[26] Witten, I. H., Frank, E., Hall, M. A., Pal, C. J., Data, M.: Practical machine learning tools and techniques, in *Data mining*, 2005, Vol. 2, No. 4, pp. 403–413.

[27] Vuković, N., Petrović, M., Miljković, Z.: A comprehensive experimental evaluation of orthogonal polynomial expanded random vector functional link neural networks for regression, *Applied Soft Computing Journal*, Vol. 70, pp. 1083–1096, 2018.

[28] Petrović, M., Jokić, A., Kulesza, Z., Miljković, Z.: Deep learning of mobile service robots, in *Book Service robots – Advances in Research and Applications*, Nova Science Publishers, New York, 2021, pp. 77–97.

[29] Krizhevsky, A., Sutskever, I., Hinton, G. E.: ImageNet Classification with Deep Convolutional Neural Networks, in *Advances in neural information processing systems (NIPS)*, 2012, pp. 1097–1105.

[30] Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition, in *3rd International Conference on Learning Representations (ICLR)*, 2015, pp. 1–14.

[31] He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.

[32] Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., Adam, H.: MobileNets: Efficient convolutional neural networks for mobile vision applications, *arXiv preprint arXiv:1704.04861*, 2017.

[33] Huang, G., Liu, Z., Van Der Maaten, L., Weinberger, K. Q.: Densely connected convolutional networks, in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4700–4708.

[34] Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S.: An image is worth 16x16 words: Transformers for image recognition at scale, *arXiv preprint arXiv:2010.11929*, 2020.

[35] Nevil, K., Priyank, V., Jenish, H., Ronak, B.: Public Datasets, Casting Product Image Data for Quality Inspection, *https://www.kaggle.com/datasets/ravirajsinh45/reallife-industrial-dataset-of-casting-product*, 2020.

[36] Møller, M. F.: A scaled conjugate gradient algorithm for fast supervised learning, *Neural networks*, Vol. 6, No. 4, pp. 525–533, 1993.

[37] Kingma, D. P., Ba, J.: Adam: A method for stochastic optimization, *arXiv preprint arXiv:1412.6980*, 2014.