Journal of Machine Engineering, 2023, Vol. 23, No. 4, 18–32
ISSN 1895-7595 (Print) ISSN 2391-8071 (Online)

Received: *31 March 2023* / Accepted: *11 October 2023* / Published online: *17 October 2023*

Industry 4.0, machining,
machine learning, monitoring

Arash EBRAHIMI ARAGHIZAD[1*], Faraz TEHRANIZADEH[1],
Kemal KILIC[2], Erhan BUDAK[1]

SMART TOOL-RELATED FAULTS MONITORING SYSTEM USING PROCESS SIMULATION-BASED MACHINE LEARNING ALGORITHMS

In this paper a novel approach for monitoring tool-related faults in milling processes by utilizing process simulation-based machine learning algorithms, specifically Random Forest algorithms, for fault detection is presented. In order to train machine learning models in tool condition monitoring, laboratory tests have traditionally been required. This method eliminates the need for costly, time-consuming laboratory tests. The training process has been simplified by utilizing analytical simulation data and provides a more cost-effective solution by leveraging analytical simulation data. Based on the results of this study, the proposed approach has been demonstrated to be 94% accurate at predicting tool-related faults, demonstrating its potential to serve as an efficient and viable alternative to conventional methods. These findings have been supported by actual measurement data, with a notable accuracy rate of 93% in the predictions. Furthermore, the results indicate that process simulation-based machine learning algorithms will have a significant impact on the tools condition monitoring and the efficiency of manufacturing processes more generally. To further enhance the capabilities of the proposed fault monitoring system, process-related and machine-related faults will be investigated in future research. Several machine learning algorithms will be explored as well as additional data sources will be integrated in order to enhance the accuracy and reliability of fault detection.

1. INTRODUCTION

These days, modern manufacturing industry are constantly striving to improve production efficiency, reduce costs, and maintain high product quality standards. Cutting tool condition is a key factor that influences these objectives in milling processes. Tool condition monitoring (TCM) is important because it helps to predict and detect tool faults and failure, resulting in reduced machine downtime, improved product quality, and more efficient tool replacement procedures. By applying machine learning algorithms for tool condition monitoring in milling processes, TCM will be more practical by utilizing data-driven techniques.

[1] Manufacturing Research Laboratory, Sabanci University, Turkey
[2] Faculty of Engineering and Natural Sciences, Sabanci University, Turkey
[*] E-mail: aebrahimi@sabanciuniv.edu
https://doi/org/10.36897/jme/174018

In subtractive manufacturing, milling is the process by which materials are removed from workpieces with rotating cutting tools in order to achieve the desired shapes and features. It is known that the performance of cutting tools deteriorates over time as a result of wear, breakage, and chipping, which can result in undesirable consequences such as increased production costs, a decrease in product quality, and a longer production cycle. In modern manufacturing, milling process simulations have become an integral part of the process, as they enable engineers to analyse and optimize various aspects of the machining process without having to conduct physical experiments. These simulations are particularly beneficial when using analytical methods for calculating cutting forces because they provide a comprehensive understanding of how the cutting tool interacts with the workpiece. A variety of factors are considered in these models, including tool geometry, workpiece material properties, cutting speed, feed rate, and depth of cut. Mechanic models, oblique cutting theories, and linear edge force models are common analytical methods. The linear edge force model is an analytical method used to predict cutting forces in milling processes. Using this simplified approach, it is possible to estimate the force experienced during machining by considering the cutting forces acting along the cutting edge of the milling tool. There are certain cases where the linear edge force model is particularly useful, such as when cutting conditions are stable and the cutting-edge engagement with the workpiece is relatively constant.

The linear edge force model [1] is based on the idea that the cutting forces acting on the tool can be represented as a linear distribution of force components along the cutting edge. These force components typically include the tangential force (Ft), radial force (Fr), and axial force (Fa). The model divides the cutting edge into small segments, and the forces acting on each segment are calculated based on the chip thickness, cutting speed, and other relevant parameters. The linear edge force model offers several advantages in milling process simulations, such as its simplicity and relatively low computational requirements. This model can provide a reasonable estimation of cutting forces, which can be used for optimizing machining parameters, tool geometries, and cutting strategies. However, it is essential to consider the limitations of the linear edge force model, such as its reliance on the assumption of constant cutting-edge engagement and stable cutting conditions.

Traditionally, TCM has been approached through various methods such as direct observation, indirect monitoring using sensors, and statistical process control. While these methods have proven useful to some extent, they often suffer from limitations such as subjectivity, low sensitivity (small changing in input parameters), and a lack of adaptability to varying cutting conditions. Implementing tool condition monitoring in milling operations can lead to several benefits, such as reduced machine downtime, improved product quality, extended tool life, and optimized tool replacement scheduling. Additionally, TCM can be integrated with Industry 4.0 technologies, such as internet of things(IoT) and cloud computing, to enable remote and centralized monitoring of multiple milling machines in a manufacturing facility. In recent years, the emergence of machine learning algorithms has opened new approaches for TCM, offering a more sophisticated and data-driven approach to the problem.

As machine learning algorithms are capable of processing large amounts of data and learning complex patterns, they are particularly suitable for TCM applications involving

multiple factors, such as cutting forces, vibrations, and acoustic emissions. In order to prevent and detect tool failure more accurately and in real-time, machine learning models are trained on simulation data from milling processes. This allows tool maintenance and replacement to be approached in a more proactive manner.

This paper will investigate various machine learning algorithms, such as Multiple Linear Regression, K-Nearest Neighbor (KNN), and Random Forest, in the context of TCM for milling processes. The performance of these algorithms has been assessed in terms of accuracy, sensitivity, and robustness, and provide recommendations for their practical implementation in industrial settings.

Typically, measuring cutting forces serves as an indirect approach for real-time tool condition monitoring [2]. The amplitude of cutting force is used in milling operations as a means of monitoring flank wear. In some studies, online tool condition monitoring systems based on cutting forces has been developed [3]. During the milling process, tool wear was monitored by measuring the average cutting force, revealing that the variation in cutting force consistently increased throughout the machining process. This observation confirmed that cutting tools progressively lost their sharp edges and became worn [4, 5]. Artificial Neural Network (ANN) based tool condition monitoring systems were developed using cutting force signals in milling processes to predict flank wear and surface roughness, indicating that cutting force signals increased alongside tool wear [6]. Tool wear in face milling was estimated using the Normalized Cutting Force (NCF) indicator and the Torque-Force Distance (TFD) indicator. The TFD method was found to be superior to NCF, as it remained unaffected by cutting parameters and their interactions [7]. An analysis of wear progression and changes in cutting force was conducted for coated carbide tools. This study illustrates that flank wear was the primary failure mode and had a significant impact on the tool's life [8]. Tool wear in the milling process was monitored using cutting force as the monitoring signal and the Continuous Hidden Markov Model (CHMM) as a diagnostic technique [9]. By tracking the tangential and radial cutting force coefficients during the end milling process, tool wear was monitored. The behaviour of these cutting force coefficients was found to be independent of cutting conditions and correlated with tool wear [10]. Tool Condition Monitoring systems (TCMs) for Glass Fiber Reinforced Plastic (GFRP) composite end milling were developed using cutting force signals and the Adaptive Network-based Fuzzy Inference System (ANFIS). The findings confirmed that ANFIS-based feed force data accurately predicted tool wear [11].

The tool condition was predicted using various decision-making algorithms based on the extracted features. Decision-making algorithms play a crucial role in the development of Tool Condition Monitoring systems (TCMs). A wide range of techniques has been explored to automate TCMs, such as Probabilistic Neural Network (PNN) [12], Support Vector Regression (SVR) [13], Support Vector Machine (SVM) [14], pattern recognition, Artificial Neural Networks (ANN) [15, 16], fuzzy logic [17, 18], and genetic algorithms [19]. More recently, researchers have employed the Hidden Markov Model [9], ANFIS [11], and Decision Trees [20] to predict tool conditions.

In this work the data mining freeware named “WEKA” was used for feature classification. Among all classifiers, the random forest classifiers yield the highest classification accuracy. As a result, this study compared tree classifiers to determine the best

among them. Other algorithms, such as the KNN algorithm, Naïve Bayes, and LWL, can also be employed if they offer superior classification accuracy. In this case, tree classifiers, including J48, Logistic Model Tree, and Random Forest algorithms are used to classify different milling cutter conditions based on a 10-fold cross-validation. Classification validation is demonstrated using a confusion matrix, as it effectively categorizes distinct tool conditions.

This study presents a novel approach to tool-related fault detection in milling processes by leveraging machine learning algorithms and simulation data. Utilizing machine learning algorithms typically requires a substantial number of tests with various cutting parameters to achieve satisfactory results, which can consume significant time and financial resources. By training machine learning algorithms with milling process simulation data, the drawbacks associated with conducting extensive cutting tests have been effectively circumvented. To accomplish this, the simulations generate a comprehensive milling process database by performing numerous simulations with varying input parameters. Subsequently, the machine learning algorithms were trained using these databases. Finally, this method can identify potential tool-related faults in milling processes. As demonstrated in the results section of this study, the proposed algorithm exhibits a high degree of accuracy in detecting various tool-related faults.

2. MILLING FORCE MODEL

In the present study, the linear edge force model [21] is employed to calculate milling forces for end-milling tools and these simulations were calibrated using measured test data and machine learning algorithms. In order to this input parameters and results of simulation are used as inputs for machine learning algorithms and the outputs of machine learning algorithms were measured data. By using this method, the accuracy of analytical simulations has been improved. To determine cutting forces for each angular increment of the tool, differential forces are computed for every axial element (i) on each tooth (j) at a specific rotational position (ϕ) throughout a complete rotation of the cutting tool:

$$\begin{gathered} \varphi_{i,j}(\phi) = \varphi_{i,j} + \phi \\ dF_r(i,j,\phi) = g\left(\varphi_{i,j}(\phi)\right)\left[K_{re} + K_{rc}(i,j)h_{i,j}(\phi)\right]dz \\ dF_t(i,j,\phi) = g\left(\varphi_{i,j}(\phi)\right)\left[K_{te} + K_{tc}(i,j)h_{i,j}(\phi)\right]dz \\ dF_a(i,j,\phi) = g\left(\varphi_{i,j}(\phi)\right)\left[K_{ac}(i,j)h_{i,j}(\phi)\right]dz \end{gathered} \tag{1}$$

Where the cutting force coefficients K_{rc}, K_{tc} and K_{ac} are calculated using the oblique cutting force model, combined with orthogonal cutting data [21], while taking into account local oblique angles ($\eta_{i,j}$) for each element. The edge force coefficients K_{re}, K_{te} and K_{ae} are typically determined from cutting tests, but can also be predicted using thermo-mechanical models applied to the third deformation zone [22]. In calculating the force coefficients, the rake angle on the cutting edges is assumed to be constant; however, it may vary along the cutting edges depending on the manufacturing process of these tools. In such cases, the local rake angle should be utilized in force coefficient calculations [21, 23].

The binary function $g(\varphi_{i,j}(\phi))$ equals 1 when the element is in cut (i.e. $\varphi_{start} \leq \varphi_{i,j}(\phi) \leq \varphi_{exit}$ and 0 otherwise. $\varphi_{i,j}(\phi)$ represents the angular position of each point on the edge when the tool's rotation angle is ϕ. dz denotes the thickness of each axial element. As depicted in Fig. , $\Delta\varphi_{i,j}$ differs for each edge at a specific axial position, and therefore, $h_{i,j}(\phi)$ (chip thickness) can be defined as follows:

$$h_{i,j}(\phi) = \frac{\Delta\varphi_{i,j}}{2\pi} Nf \sin\left(\varphi_{i,j}(\phi)\right) \tag{2}$$

where *N* represents the number of teeth and *f* corresponds to the nominal feed per tooth.

The total forces in x, y, z directions for angular orientation of the tool can be obtained by summation of the elemental differential forces:

$$\begin{aligned} F_x(\phi) &= \sum_{i=0}^{a}\sum_{j=1}^{N_t}\left[-dF_r(i,j,\phi)\, sin\left(\varphi_{i,j}(\phi)\right) - dF_t(i,j,\phi)\, cos\left(\varphi_{i,j}(\phi)\right)\right] \\ F_y(\phi) &= \sum_{i=0}^{a}\sum_{j=1}^{N_t}\left[-dF_r(i,j,\phi)\, cos\left(\varphi_{i,j}(\phi)\right) + dF_t(i,j,\phi)\, sin\left(\varphi_{i,j}(\phi)\right)\right] \\ F_z(\phi) &= \sum_{i=0}^{a}\sum_{j=1}^{N_t} dF_a(i,j,\phi) \end{aligned} \tag{3}$$

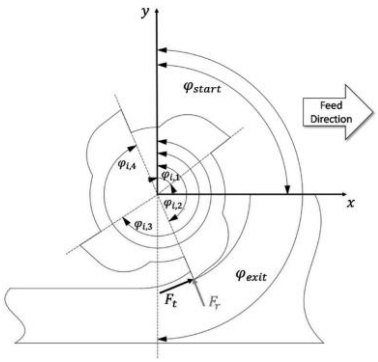

Fig. 1. The schematic view of the milling cutting force directions [24]

3. APPLIED MACHINE LEARNING FOR FAULT DETECTION

Since fault diagnosis is the most challenging aspect of process and machine repair, the majority of downtime is spent localizing the fault rather than addressing it [25]. Consequently, organizations are exploring innovative methods to enhance the root cause analysis (RCA) process for faults.

As a first step, it is essential to define the problem and determine the appropriate data analytics techniques. In order to make the required data suitable for further analysis, it is necessary to collect, and preprocess the data based on the problem and the selected method.

As a result, developing, and evaluating a data model is vital. To resolve the issue, the outcomes are examined. The process is typically repeated several times in order to achieve better results.

Based on the type of input data and the type of learning system, machine learning algorithms can be divided into three categories. In supervised learning, algorithms are trained to map inputs to known outputs (provided by experts). In unsupervised learning, models or functions are developed without incorporating any previously known outputs. This approach typically analyses large datasets to discover meaningful patterns or classifications. Additionally, reinforcement learning allows a machine to determine its performance based on a reward signal that has been previously defined.

Two major objectives of these algorithms are to classify or cluster data, and to identify a trend or relationship over time, respectively. This study investigated a number of machine learning algorithms for data training and fault detection and selected three algorithms based on the results: Multiple Linear Regression, K-Nearest Neighbor (KNN), and Random Forest.

Multiple Linear Regression (MLR): Multiple linear regression is a statistical method used to model the relationship between a dependent variable and two or more independent variables. It is an extension of simple linear regression, which involves only one independent variable. The primary goal of multiple linear regression is to create a predictive model that can estimate the value of the dependent variable based on the values of the independent variables. The multiple linear regression model takes the form:

$$y = \beta_0 + \beta_1 * x_1 + \beta_2 * x_2 + ... + \beta_n * x_n + \varepsilon \quad (4)$$

Here, y represents the dependent variable, $x_1, x_2, ..., x_n$ are the independent variables, β_0 is the intercept, $\beta_1, \beta_2, ..., \beta_n$ are the regression coefficients, and ε is the residual or error term, which accounts for the variation in the data not explained by the model.

The regression coefficients ($\beta_1, \beta_2, ..., \beta_n$) represent the change in the dependent variable associated with a one-unit change in the corresponding independent variable, while holding all other variables constant. These coefficients are estimated using a technique called ordinary least squares (OLS), which minimizes the sum of the squared differences between the observed values of the dependent variable and the values predicted by the model.

K-Nearest Neighbor (KNN): K-Nearest Neighbor (KNN) is a non-parametric, instance-based, supervised learning algorithm used for classification and regression tasks. It is considered one of the simplest and most intuitive machine learning algorithms, owing to its easy-to-understand approach and minimal training requirements. In KNN, the prediction for a new data point is determined based on the K closest data points (neighbours) from the training dataset. The algorithm operates under the assumption that similar data points are located near each other in the feature space. Euclidean distance is a commonly used distance metric in the KNN algorithm. It is the straight-line distance between two points in an n-dimensional space [26, 27]. The algorithm calculates the Euclidean distance between the new data point and every point in the training dataset, then sorts these distances in ascending order.

The choice of K is crucial in KNN, as it directly affects the algorithm's performance. A small value of K might result in overfitting, while a large value of K can lead to underfitting. Typically, the optimal value of K is determined through techniques like cross-validation. KNN is sensitive to the scale of the input features, so it is often necessary to normalize or standardize the data before applying the algorithm. Furthermore, KNN is sensitive to the

presence of irrelevant or noisy features, which can negatively impact its performance. Feature selection techniques can help mitigate this issue.

Below is a simple pseudocode for the K-Nearest Neighbor (KNN) algorithm for classification:

- Function KNN_Classify (new_data_point, training_data, K):
- Initialize an empty list called distances_list.
- For each data_point in training_data:
- Calculate the distance (e.g., Euclidean distance) between new_data_point and data point.
- Add (distance, data_point's class) to the distances_list.
- End For
- Sort distances_list in ascending order of distance
- Select the first K elements from the sorted distances_list.
- Initialize an empty dictionary called class_votes.
- For each (distance, class) in the K selected elements:
- If class is not in class_votes:
- class_votes[class] = 0
- End If
- class_votes[class] += 1
- End For
- Determine the class with the highest vote in class_votes.
- Return the class with the highest vote as the prediction for new_data_point.

Random Forest (RF): Random Forest is an ensemble learning method used for both classification and regression tasks. It operates by constructing multiple decision trees during the training phase and then aggregating their outputs to make a final prediction. The main idea behind the Random Forest algorithm is to combine the results of multiple weak learners (decision trees) to obtain a more accurate and robust model. It is particularly effective for handling high-dimensional data and can address classification and regression problems. Here's the logic behind how Random Forest works:

1. Bootstrapping: For a given dataset, Random Forest creates multiple bootstrap samples by randomly selecting data points with replacement. Each bootstrap sample is used to train a separate decision tree.
2. Feature Randomness: During the process of growing individual decision trees, at each node, a random subset of features is selected to determine the best split. This random feature selection introduces diversity among the trees and reduces the correlation between them.
3. Decision Tree Construction: A decision tree is built for each bootstrap sample using the selected features at each node. The tree construction continues until a maximum depth is reached or a minimum number of samples per leaf is obtained.
4. Aggregating Predictions: Once all decision trees are constructed, the Random Forest algorithm makes a prediction by combining the predictions of all trees. For classification problems, this is typically done by taking the majority vote among the tree predictions. For regression problems, the average prediction of all trees is used.

In addition to reducing overfitting risk, the Random Forest algorithm also improves model generalization by combining predictions from multiple decision trees. Furthermore, this algorithm is robust to noise and can handle a large dataset, making it an excellent choice for a variety of machine learning applications.

Here's a pseudocode representation of the Random Forest algorithm:

Procedure Random_Forest (training_data, num_trees, max_depth, min_samples_leaf, max_features):

- Initialize an empty list called forest.
- For i = 1 to num_trees:
  a. Bootstrap_sample = Create_Bootstrap_Sample(training_data)
  b. Tree = Build_Decision_Tree (Bootstrap_sample, max_depth, min_samples_leaf, max_features)
  c. Add Tree to forest.
- Return forest.

Procedure Create_Bootstrap_Sample(data):

- Randomly select data points with replacement from the given data.
- Return the created bootstrap sample.

Procedure Build_Decision_Tree (data, max_depth, min_samples_leaf, max_features):

- If max_depth is reached or the number of samples in data is less than or equal to min_samples_leaf:

a. Return a leaf node with the majority class (classification) or average value (regression) of data.

- Randomly select a subset of features up to max_features.
- Determine the best split using the selected features.
- Split the data into left and right subsets based on the best split.
- left_child = Build_Decision_Tree (left_subset, max_depth, min_samples_leaf, max_features)
- right_child = Build_Decision_Tree (right_subset, max_depth, min_samples_leaf, max_features)
- Return a decision node with the best split and left_child, right_child as its children.

Procedure Predict (forest, test_data):

- Initialize an empty list called predictions.
- For each test_point in test_data:
  a. tree_predictions = [tree. predict (test_point) for tree in forest]
  b. prediction = Majority_Vote (tree_predictions) (classification) or Mean (tree_predictions) (regression)
  c. Add prediction to predictions.
- Return predictions.

3.1. DATASET FOR TRAINING MACHINE LEARNING ALGORITHMS

As a result of the application of an algorithm based on process simulation, significant advancements have been made in the monitoring of tool condition. Tool-related faults can be

detected more effectively and economically by eliminating extensive laboratory testing. As a result of this research and development, this approach can help tool condition monitoring to be more practical and enhance manufacturing processes in general. As mentioned before, in order to predict by machine learning, the algorithms must train by the acceptable portion of datasets. A common practice is to split the dataset into the following portions:

1. Training set: This subset is used for training the machine learning model. The model learns the patterns and relationships within the data. A typical proportion for the training set is around 70%–80% of the entire dataset.
2. Test set: This subset is used to evaluate the performance of the trained model. It is crucial that the test set is separate from the training set and is not used during the training process. The test set typically comprises 20%–30% of the entire dataset.

3.2. NPUTS AND OUTPUTS OF TRAINED DATASET PARAMETERS

To perform a milling simulation, several inputs are required to represent the milling process accurately. Some of the essential inputs include:

- Tool geometry: This includes the tool's radius, number of teeth, helix angle, rake angle, and cutting-edge geometry. These parameters are crucial for simulating the tool's interaction with the workpiece.
- Workpiece material: The material properties of the workpiece and cutting force coefficients during the milling process.
- Cutting conditions: These include cutting parameters such as spindle speed, feed rate, and depth of cut, which affect the cutting forces.
- Cutting tool material: The cutting tool material affects the tool's performance, wear resistance, and cutting forces. Common cutting tool materials include high-speed steel (HSS), carbide, and polycrystalline diamond (PCD).
- Coolant and lubrication: The type and application method of coolant or lubricant used in the milling process impact the tool's temperature, cutting forces, and chip formation.

The input parameters employed for constructing the database in this study are follows:

- Tool diameter: 10, 14, 18 and 20 mm.
- Teeth number: 3,4, and 6.
- Tool helix angle: 30, 35, and 45°.
- Tool rake angle: 5, 7 and 11°.
- Spindle speed: 4000, 6000 and 8000 rpm.
- Feed per revolution per tooth: 0.1, 0.3, 0.5, 0.7, 0.9 and 1. mm/(rev*tooth).
- Axial depth of cut: 1, 3, 5, 7, 9 and 10 mm.
- Radial depth of cut: 0.4, 1.6, 2, 2.4, 2.8, 3.6 and 4 mm.

Utilizing these input parameters, a total of 81,648 simulations were conducted.

This study evaluated a number of machine learning algorithms. Root-mean square error (RMSE) and mean absolute error (MAE) are two commonly used metrics for evaluating such algorithms.

- Root Mean Squared Error (RMSE):

RMSE is a measure of how well the machine learning model fits the data, similar to MSE. However, RMSE is the square root of the average squared difference between the predicted values and the actual values, which makes it more interpretable in the same units as the target variable. The equation for RMSE is:

$$\mathrm{RMSE} = \sqrt{\frac{\sum_{i=1}^{n} (y_{pred} - y_{actual})^2}{n}} \tag{5}$$

Where n is the number of observations in the test set, y_pred is the predicted value of the target variable and y_actual is the actual value of the target variable. A lower RMSE indicates that the model is better at predicting the target variable. However, it is also sensitive to outliers in the data.

- Mean Absolute Error (MAE):

MAE is another measure of how well the machine learning model fits the data. It is the average absolute difference between the predicted values and the actual values. The formula for MAE is:

$$\mathrm{MAE} = \frac{\sum_{i=1}^{n} \left| y_{pred} - y_{actual} \right|}{n} \tag{6}$$

MAE is less sensitive to outliers than RMSE because it does not square the differences between the predicted values and the actual values. However, it may not be as interpretable as RMSE because it is not in the same units as the target variable.

Table 1. Comparison of various ML algorithms

	Random Forest	Random Tree	KNN	MLR
RMSE	0.1070	0.1852	0.1078	1.4068
MAE	0.0426	0.0313	0.0462	1.1891

As previously discussed, the training-data utilized for the development of the machine learning models in this study consisted of 65% of the simulation data, while the remaining 35% was reserved as test-data. As depicted in Table 1, the Random Forest algorithm demonstrated superior performance compared to other machine learning methods. Consequently, this algorithm will be employed as the primary fault detection mechanism in the process.

4. RESULTS AND DISCUSSION

By using machine learning algorithms, this method can gain several key insights into the detection of tool-related faults in milling processes. As described above the Random forest algorithm has been used in order to detect tool-related faults in milling processes. It has been tested 50 different scenarios for each input parameter in order to determine whether or not

this method is accurate for vast range of input parameters. Approximately 50 simulations were conducted in order to determine how well the algorithm identified faults associated with the tool radius, as an example. These simulations involved deliberately changing the radius, teeth number, helix and rake angle of the tool in the simulation, as well as executing machine learning models in order to test how well the method would perform for varying parameters related to the tool. The results of this method are presented in Fig. 2:

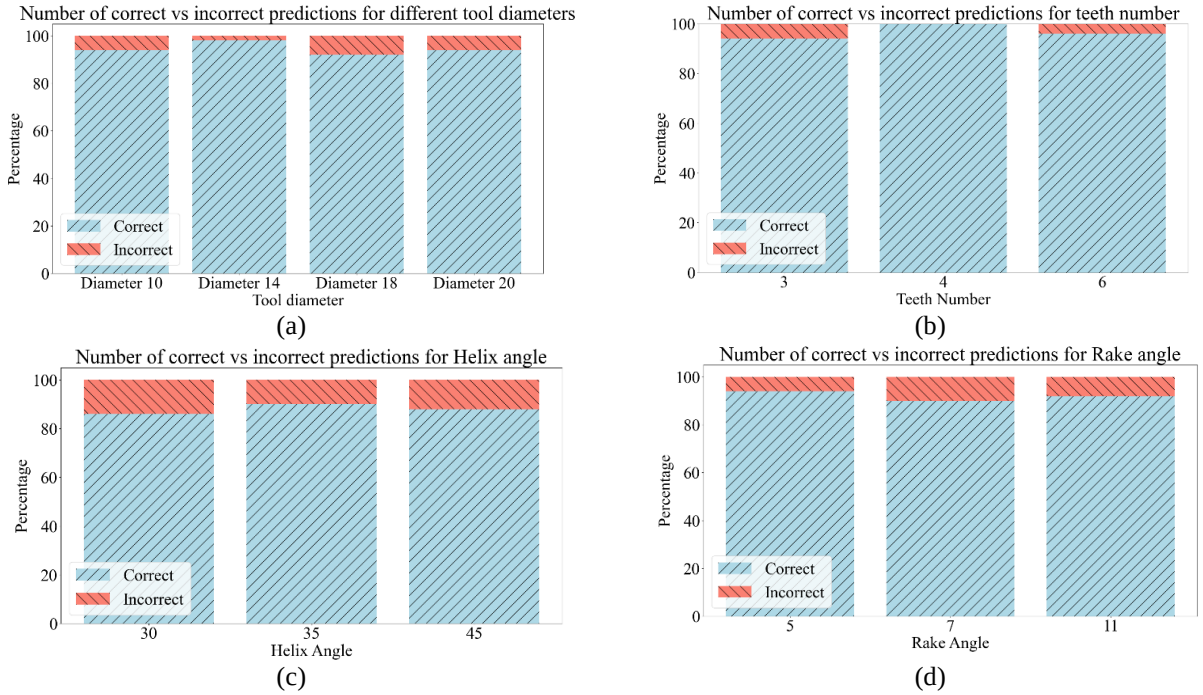

Fig. 2. Correct vs incorrect fault detection of (a) Tool diameter, (b) Teeth number, (c) Tool helix angle and (d)Tool rake angle

Figure 2. illustrates the percentage of correct and incorrect prediction by using mentioned algorithms in various situations. For instance, Fig. 2a first column shows the percentage of correct prediction of tool diameter which was 10 mm. Moreover, Fig. 2a. The first column shows that the algorithm can predict the cutting tool diameter with 10(mm) can be predicted with 96% accuracy. Each column of every graph shows the correct prediction percentage of that parameter. It is evident from Fig. 2 that the tool parameters that have an enormous impact on the cutting force have been identified with higher precision.

The figure above indicates that the number of incorrect predictions for helix and rake angle is higher than for the other two parameters (e.g. tool diameter and tooth number). This is because helix and rake angle appear to have less impact on cutting forces compared to the tool diameter and tooth number. Aside from that, all of the tool related faults detected within acceptable precision.

The cutting forces were conducted using Piezo-electric Dynamometer 9257BA, amplifier and data acquisition NI USB-6259. 16 and 20 mm Solid Carbide end mill cutter with 4 flutes and 7075-T6 aluminum and 1050 steel were used as workpieces during the tests.

Fig. 3. Measurement setup

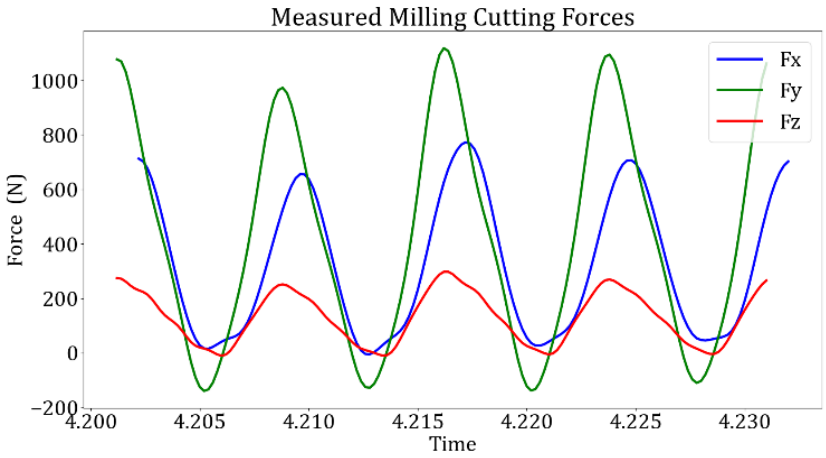

Fig. 4. One revolution of measured cutting forces

Figure 4 illustrates the measured cutting forces with dynamometer. The cutting parameters which have been used for conducting this test are spindle speed is 2000 rpm, Feed is 1600 mm/min, axial depth is 4(mm), radial depth of cut is 4 mm and the workpiece material is 1050 steel. This measured data has been used for input of proposed algorithm and the ML algorithm can predict that the tool diameter is 16 mm with 4 cutting flutes, helix angle is 35° and rake angle is 7°. The predicted tool geometry parameters are the same as the tool which has been used for conducting this test.

Figure 5 shows the other measured cutting forces. In this test spindle speed is 1000 rpm, Feed is 1000 mm/min, axial depth is 7 mm, radial depth of cut is 2 mm and the workpiece material is 1050 steel. The proposed algorithm predict that tool diameter is 20 mm with 4 cutting flutes, helix angle is 30° and rake angle is 5°. The predicted tool geometry parameters are the same as the tool which has been used for conducting this test.

15 tests have been conducted to assess the accuracy of the proposed algorithm, which includes four parameters of tool geometry: diameter, tooth number, helix, and rake angle of the tool. The algorithm yielded 56 correct predictions of tool geometry parameters, showcasing an impressive 93% accuracy by utilizing real measured data.

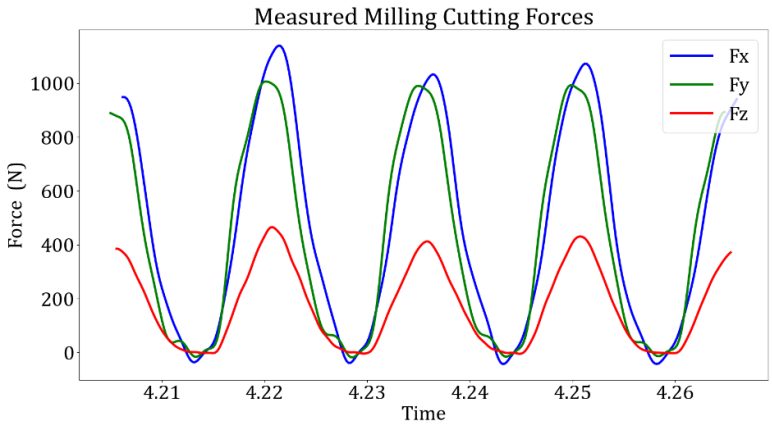

Fig. 5. One revolution of measured cutting forces.

5. CONCLUSION

The main goal of this paper has been to show how using process simulation-based machine learning algorithms can effectively monitor and detect tool-related faults during milling processes. The approach proposed in this study eliminates the need for expensive and time-consuming lab tests by training machine learning models with milling process simulation data.

Throughout this investigation, a range of machine learning algorithms underwent testing using two evaluation metrics. Notably, the random forest algorithm demonstrated superior performance when contrasted with alternative methods for handling such input and output parameters.

Based on the outcomes of this research, the algorithm is capable of achieving a 94 percent accurate prediction rate for tool-related faults. Attaining such a high level of accuracy in predicting tool-related faults solely through the utilization of simulation data can enhance the viability of monitoring systems. This method holds the potential to eliminate the necessity for a substantial volume of tests, a common requirement in typical monitoring applications. These findings have been supported by actual measurement data, with a notable accuracy rate of 93 percent in the predictions.

Although the results are promising, it is essential to continue with further research and improvements. To elevate the overall performance and reliability of the fault monitoring system, future endeavours could delve into integrating supplementary data sources and adopting advanced machine learning algorithms. Furthermore, the scope of this work can be expanded to encompass process and machine-related faults in forthcoming studies.

This paper has resulted in a notable stride forward in the realm of tool condition monitoring through the application of a process simulation-based algorithm. By streamlining the detection of tool-related faults in a more efficient and cost-effective manner, the demand for extensive laboratory testing has been minimized. This approach holds the capacity to bring about a transformative impact on the domain of tool condition monitoring, thereby enhancing manufacturing processes as a cohesive whole, all the while fostering ongoing research and development efforts.

REFERENCES

[1] BUDAK E., ALTINTAS Y., ARMAREGO E.J.A., 1996, *Prediction of Milling Force Coefficients from Orthogonal Cutting Data*, J. Manuf. Sci. Eng., 118/2, 216–224, https://doi.org/10.1115/1.2831014.

[2] BALAZINSKI M., CZOGALA E., JEMIELNIAK K., LESKI J., 2002, *Tool Condition Monitoring Using Artificial Intelligence Methods*, Eng. Appl. Artif. Intell., 15, 73–80.

[3] ELBESTAWI M.A., PAPAZAFIRIOU T.A., DU R.X., 1991, *In-Process Monitoring of Tool Wear in Milling Using Cutting Force Signature*, Int. J. Mach. Tools Manuf., 31, 55–73.

[4] TANSEL I.N., ARKAN T.T., BAO W.Y., MAHENDRAKAR N., SHISLER B., SMITH D., Mc COOL M., 2000, *Tool Wear Estimation in Micro-Machining*, Part I: *Tool Usage–Cutting Force Relationship*, Int. J. Mach. Tools Manuf., 40, 599–608.

[5] TANSEL I.N., ARKAN T.T., BAO W.Y., MAHENDRAKAR N., SHISLER B., SMITH D., McCOOL M., 2000, *Tool Wear Estimation in Micro-Machining*, Part II: *Neural-Network-Based Periodic Inspector for Non-Metals*, Int. J. Mach. Tools Manuf., 40, 609–620.

[6] SAGLAM H., UNUVAR A., 2003, *Tool Condition Monitoring in Milling Based on Cutting Forces by a Neural Network*, Int. J. Prod. Res. 41, 1519–1532.

[7] KULJANIC E., SORTINO M., *2005, Twem, a Method Based on Cutting Forces—Monitoring Tool Wear in Face Milling*, Int. J. Mach. Tools Manuf., 45, 29–34.

[8] LI H.Z., ZENG H., CHEN X.Q., 2006, *An Experimental Study of Tool Wear and Cutting Force Variation in the end Milling of Inconel 718 with Coated Carbide Inserts*, J. Mater. Process. Technol., 180, 296–304, https://doi.org/ 10.1016/j. jmatprotec.2006.07.009.

[9] WANG M., WANG J., 2012, *CHMM for Tool Condition Monitoring and Remaining Useful Life Prediction*, Int. J. Adv. Manuf. Technol., 59, 463–471.

[10] NOURI M., FUSSELL B.K., ZINITI B.L., LINDER E., 2015, *Real-Time Tool Wear Monitoring in Milling Using a Cutting Condition Independent Method*, Int. J. Mach. Tools Manuf., 89, 1–13.

[11] AZMI A.I., 2015, *Monitoring of Tool Wear Using Measured Machining Forces and Neuro-Fuzzy Modelling Approaches During Machining of GFRP Composites*, Adv. Eng. Softw., 82, 53–64.

[12] DA SILVA R.H.L., DA SILVA M.B., Hassui A., 2016, *A Probabilistic Neural Network Applied in Monitoring Tool Wear in the end Milling Operation via Acoustic Emission and Cutting Power Signals*, Mach. Sci. Technol., 20, 386–405.

[13] SMOLA A.J., SCHÖLKOPF B., 2004, *A Tutorial On Support Vector Regression*, Statistics and computing, 14, 199–222.

[14] BENKEDJOUH T., MEDJAHER K., ZERHOUNI N., RECHAK S., 2015, *Health Assessment and Life Prediction of Cutting Tools Based on Support Vector Regression*, J. Intell. Manuf., 26, 213–223.

[15] HONG G.S., RAHMAN M., ZHOU Q., 1996, *Using Neural Network for Tool Condition Monitoring Based on Wavelet Decomposition*, Int. J. Mach. Tools Manuf., 36, 551–566.

[16] SHANKAR S., MOHANRAJ T., PRAMANIK A., 2019, *Tool Condition Monitoring While Using Vegetable Based Cutting Fluids During Milling of Inconel 625*, J. Adv. Manuf. Syst., 18, 563–581.

[17] LI X., LI H.-X., GUAN X.-P., DU R., 2004, *Fuzzy Estimation of Feed-Cutting Force from Current Measurement-a Case Study on Intelligent Tool Wear Condition Monitoring*, IEEE Trans. Syst. Man, Cybern., Part C Applications Rev., 34/4, 506–512.

[18] SHANKAR S., MOHANRAJ T., 2015, *Tool Condition Monitoring in Milling Using Sensor Fusion Technique*, Proc. Malaysian Int. Tribol. Conf., 322–323.

[19] KAYA B., OYSU C., ERTUNC H.M., OCAK H., 2012, *A Support Vector Machine-Based online Tool Condition Monitoring for Milling Using Sensor Fusion and a Genetic Algorithm*, Proc. Inst. Mech. Eng., Part B, J. Eng. Manuf., 226, 1808–1818.

[20] ELANGOVAN M., RAMACHANDRAN K.I., SUGUMARAN V., 2010, *Studies on Bayes Classifier for Condition Monitoring of Single Point Carbide Tipped Tool Based on Statistical and Histogram Features*, Expert Syst. Appl., 37, 2059–2065.

[21] TEHRANIZADEH F., KOCA R., BUDAK E., 2019, *Investigating Effects of Serration Geometry on Milling Forces and Chatter Stability For Their Optimal Selection*, Int. J. Mach. Tools Manuf., 144, 103425.

[22] BUDAK E., OZLU E., BAKIOGLU H., BARZEGAR Z., 2016, *Thermo-Mechanical Modeling of the Third Deformation Zone in Machining for Prediction of Cutting Forces*, CIRP Ann., 65, 121–124.

[23] ÖZLÜ E., EBRAHIMI ARAGHIZAD A., BUDAK E., 2020, *Broaching Tool Design Through Force Modelling and Process Simulation*, CIRP Ann., https://doi.org/10.1016/j.cirp.2020.04.035.

[24] TEHRANIZADEH F., BERENJI K.R., BUDAK E., 2021, *Dynamics and Chatter Stability of Crest-Cut end Mills*, Int. J. Mach. Tools Manuf., 171, 103813.

[25] KEGG R.L., 1984, *One-Line Machine and Process Diagnostics*, CIRP Ann., 33, 469–473.
[26] ARMAREGO E.J.A., SMITH A.J.R., GONG Z.J., 1990, *Four Plane Facet Point Drills – Basic Design and Cutting Model Predictions*, CIRP Ann., 39, 41–45.
[27] HO T.K., 1995, *Random Decision Forests,* Proc. 3rd Int. Conf. Doc. Anal. Recognit., IEEE, 278–282.