COBEM 2023 27TH INTERNATIONAL CONGRESS OF MECHANICAL ENGENEERING

ABCM Brazilian Society of Mechanical Sciences and Engineering
27th ABCM International Congress of Mechanical Engineering
December 04-08, 2023, Florianópolis-SC, Brazil

COB-2023-0853
SEMI-SUPERVISED MACHINE LEARNING FOR CHATTER DETECTION IN TURNING WITH DATA AUGMENTATION

Ana Julia da Silva de Oliveira
Daniel Awada Elarrat Canto
Giuliana Sardi Venter
Federal University of Paraná
ana.silva05@ufpr.br
daniel.awada@ufpr.br
giuliana.venter@ufpr.br

Abstract. *Chatter occurrence in machining can lead to component failure, production delays, and increased costs. To prevent these issues, interrupting machining before chatter occurs is essential. Hence, this project pursues to create an online tool for identifying the onset of chatter enabling an automatic interruption, aiming to reduce the consequences of chatter in turning and prevent waste and breakage. Machine Learning algorithms can be used to predict and identify chatter, but they require a large amount of experimental data. Finding a suitable dataset can be challenging, and Data Augmentation may be necessary to increase the dataset size and robustness. In this study, the authors used a publicly available dataset consisting of 115 files with time series of accelerometer data, in which two uniaxial accelerometers were mounted on the tool holder itself, and a triaxial accelerometer was mounted on the base of the tool holder. The placement allowed for the measurement of vibration and acceleration during cutting tests. Data Augmentation using Windowing and Noise was applied to the dataset, increasing the number of data entry from the initial 115 to 728. The initial files were labeled using four classes based on the existence of chatter: "chatter", "without chatter", "intermediate" and "unknown". To effectively monitor and interrupt the process, it is ideal to use only two classes: "chatter" and "without chatter". Moreover, in the process of Data Augmentation, each time series entry was separated into four and their labels could be inconsistent due to the transient nature at the beginning of chatter, meaning that one entry could have both 'without chatter' and 'chatter' when separated. Therefore, a semi-supervised method for labeling through clustering was used to effectively separate the data into the desired two classes. Mathematical features both in the time and frequency domains were used for training and testing. Multiple Machine Learning algorithms were tested for efficiency and accuracy. GridSearchCV was used to find the optimized hyperparameters for all models. Results of accuracy as high as 98% with 10-fold cross-validation were obtained with Random Forest. Overall, this study provides insights into the use of Machine Learning algorithms for identifying and preventing chatter in machining, using Data Augmentation and Noise to increase the dataset size and quality. The results show promising potential for the use of relatively small datasets with missing labels, enabling a more efficient way of creating new monitoring tools for chatter.*

Keywords: *Machine Learning, Turning, Chatter, Data Augmentation, Machining*

1. INTRODUCTION

Machining lathes are used in a wide range of industries. The ability to produce parts with high precision and efficiency has made them widespread machinery of modern manufacturing. However, as all machines, machining lathes can cause failures that can compromise the performance, efficiency, and the quality of the final product.

The phenomenon known as "chatter" poses a significant challenge in machining operations. Chatter is an unwanted vibration that causes Noise during operation, compromising the surface quality of the machined material and, in extreme cases, potentially damaging the workpiece, the tool, or the machine itself. It occurs due to the friction between the material removal tool and the material being machined. The appearance of chatter not only decreases process efficiency but also increases production costs due to the need to replace damaged parts or remachining (Quintana and Ciurana, 2011).

The primary impact of chatter lies in the production interruption caused by its occurrence. These interruptions prevent the need for reworking the workpiece; however, they are costly for the company. Therefore, identifying chatter before it occurs can lead to significant savings during the manufacturing process (Sestito *et al.*, 2022).

Predictive maintenance is an alternative approach to corrective maintenance aimed at reducing machine downtime and maintenance costs. Implementing this technology requires a large amount of data and sensors that enable component monitoring. Thus, it becomes possible to implement preventive maintenance using Machine Learning. The predictive approach based on Machine Learning analyzes real-time data and seeks to identify correlations between the collected data to predict system failures or schedule maintenance of equipment (Paul *et al.*, 2022).

In this context, Machine Learning emerges as a powerful tool to facilitate the implementation of predictive maintenance. Research has already been conducted in this area; for example, (Oleaga *et al.*, 2018) presents a study on the implementation of Machine Learning for chatter detection in milling machines, using sensors at the tool center point. In (Sestito *et al.*, 2022), the authors researched the use of Machine Learning for micro-milling processes with acoustic emission data, employing in-time classification models, demonstrating the feasibility of implementing Machine Learning in online operations.

Currently, research conducted on this subject typically focuses on two or three Machine Learning models. For instance, (Oleaga *et al.*, 2018) focused on regression analysis, Artificial Neural Networks (ANN), and Decision Tree. Classification models such as Support Vector Machines (SVM), Decision Trees, Random Forests, K-Nearest Neighbors (KNN), and ANN are well-established in Machine Learning and have shown great promise in early identification of patterns and anomalies that may indicate the occurrence of failures. This study aims to compare these classification models to determine the most efficient one for chatter detection.

These models require a significant amount of data to be trained, and given the common challenge of low data volume in industrial applications, the use of Data Augmentation techniques emerges as an effective strategy to improve the robustness of Machine Learning models. This method is employed in this study, along with semi-supervised clustering, to enable and enhance the results of the chosen Machine Learning models. The selected methods for Data Augmentation are Windowing and Gaussian Noise, supported by the review (Wen *et al.*, 2021).

This article proposes a new method for identifying chatter in rotating machines using Machine Learning classification models. The objective is to develop a real-time monitoring system capable of detecting the onset of chatter, enabling predictive maintenance interventions and minimizing the negative impacts of this phenomenon on production. Through this approach, we aim to improve the efficiency of machining operations and the quality of products while simultaneously reducing costs associated with maintenance and part replacement.

2. METHODS

2.1 Overview

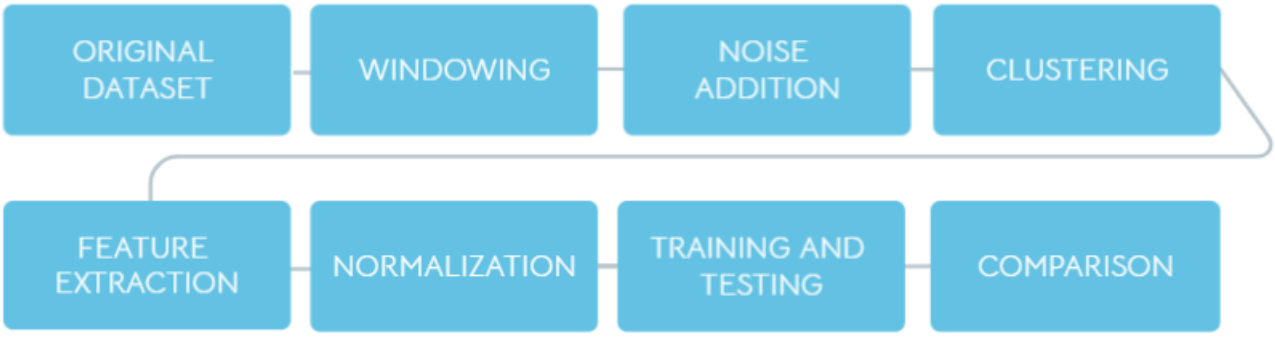

Figure 1. Visualization of the methodology of this project.

Figure 1 shows an overview of the methodology that will be discussed in further detail in this section.

2.2 Dataset

The dataset used was provided by (Khasawneh *et al.*, 2019), consisting of pre-processed and non-pre-processed data. The data used for this project were the pre-processed ones, which were collected from two uniaxial accelerometers on the tool holder and a triaxial accelerometer on the base of the tool holder. The data provided represented the acceleration of the tool holder in a time series. The files were named based on the classification of chatter presence, which could be "chatter", "no chatter", "intermediate", or "unknown." The "unknown" data were discarded, and the intermediate data were reclassified later as either "chatter" or "no chatter."

2.3 Data Augumentation

Some of the chosen models for evaluation are complex and require large amounts of data to achieve optimal results, such as ANN. The need for a large dataset led to the ideal use of Data Augmentation to expand the available data. Two Data Augmentation methods were chosen: Windowing and Gaussian Noise addition. Both methods were applied to a large publicly available dataset published by (Khasawneh *et al.*, 2019).

The Data Augmentation techniques were chosen for this project based on (Wen *et al.*, 2021). Windowing is a Data Augmentation technique where input data is divided into smaller windows or segments. This technique is used in Machine

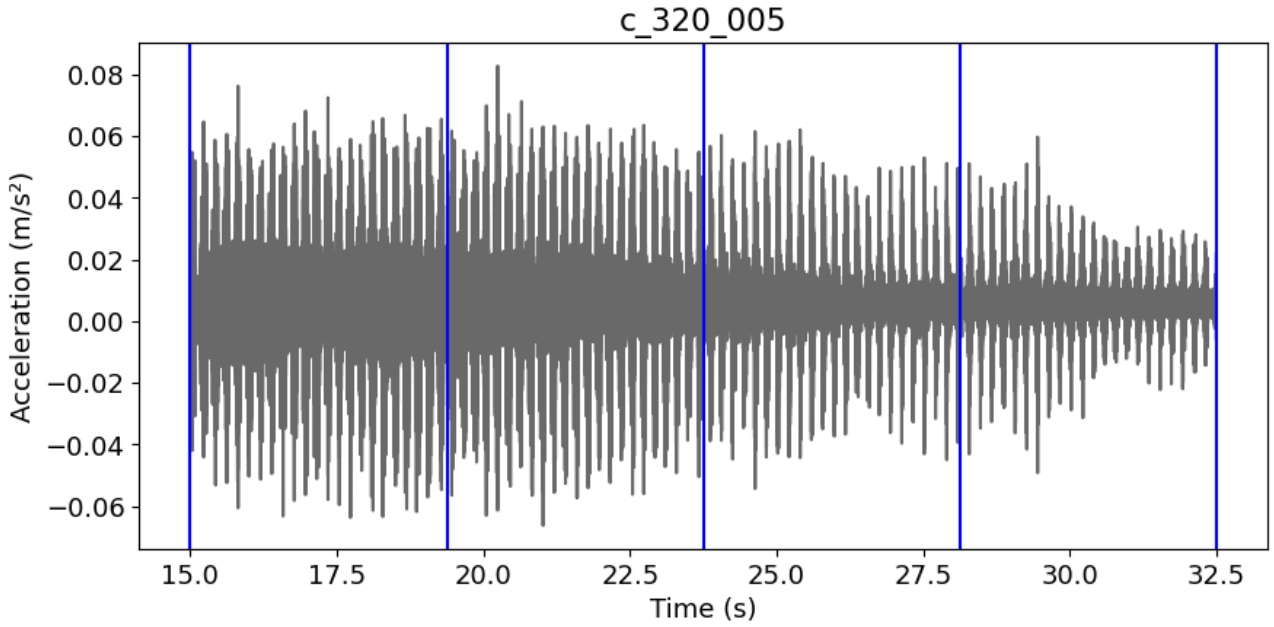

Figure 2. Visualization of splits.

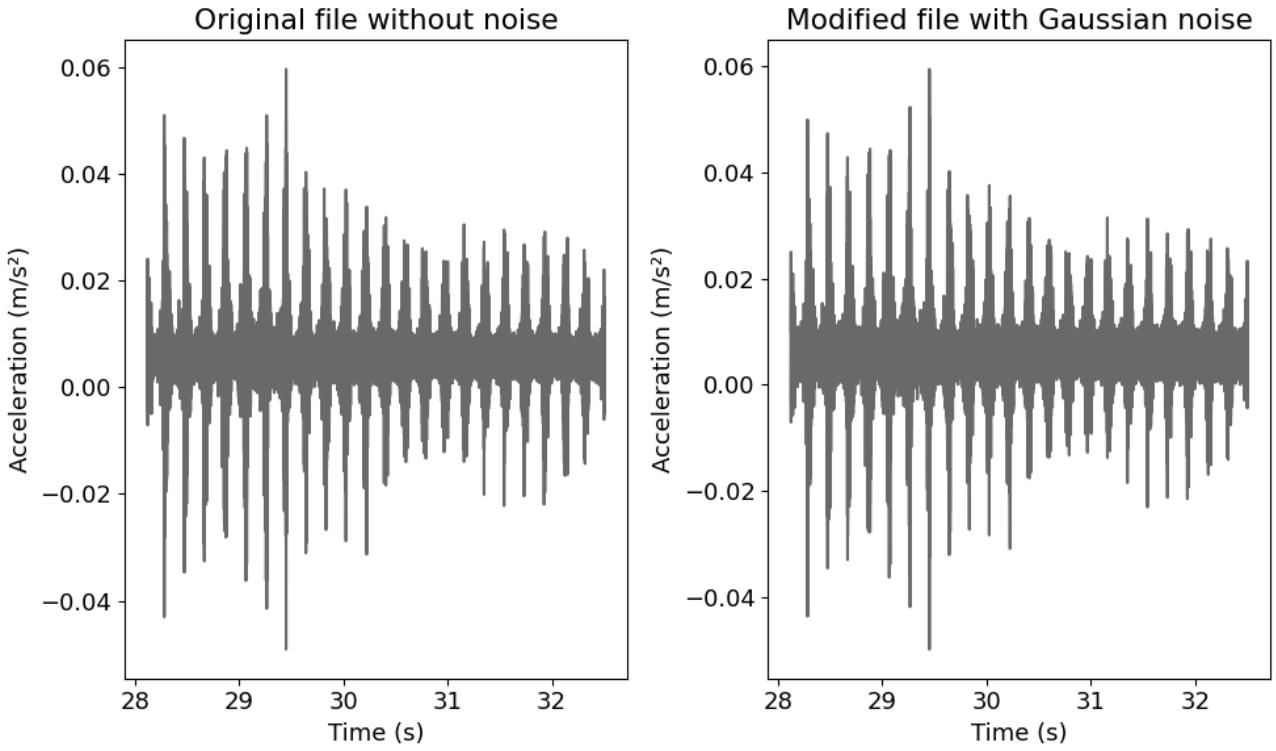

Figure 3. Gaussian Noise added to split files.

Learning to process data on time domain, which is compatible with the dataset used in this project. Windowing allows for extracting more detailed information and emphasizing short-duration patterns present in the data. This enables the model to learn more and improve its performance in chatter detection. However, the main objective of this project to use Windowing was to multiply the input data.

Each original file had a large number of data points. Thus, it was possible to obtain four new files from the original file by performing Windowing, effectively multiplying the number of files by four. Figure 2 shows how this Windowing was done: each file was divided into four new files with equal numbers of data points.

The second Data Augmentation method used was the addition of Gaussian Noise to the files. Gaussian Noise is a form of data distortion used in Data Augmentation techniques in Machine Learning. This type of Noise follows a

Gaussian distribution, where values are randomly sampled according to the specified mean and standard deviation. In this project, Gaussian Noise was used to simulate disturbances in the production line, such as vibration or acoustic signals, mimicking the interferences that may occur during the actual operation of the machines. Additionally, Gaussian Noise can help prevent overfitting by reducing the model's sensitivity to small variations in the input data and promoting improved generalization.

The amount of Noise added was discrete, as observed in Fig. 3. The Noise was proportional to the acceleration, and its exact amount was random, set uniformly through all the data, and with maximum amplitude of 10% of the acceleration at the respective data point. This amount of Noise, however, is sufficient to simulate real interferences that can disrupt the accuracy of the sensors in a real application of the project. The decision was made to limit the Noise to 10% because higher values could introduce bias to the models by giving excessive significance to the added Noise.

2.4 KMeans

KMeans is a Machine Learning tool used for data clustering. There are both unsupervised and semi-supervised algorithms for KMeans. In the semi-supervised approach, KMeans is combined with labeled data to assign labels to the clusters, using partial label information to improve classification accuracy. This allows for reducing the need to manually label large datasets, saving time and effort during the model training phase (Ahmed *et al.*, 2020). The implementation of this technique helped solve the problem of splitting generating some files whose chatter presence became unknown.

The files were originally labeled for the presence of chatter as "chatter," "without chatter," "intermediate," and "unknown." The "unknown" classified data were not used in this research, and the "intermediate" class was reclassified as either "chatter" or "without chatter." This "intermediate" class was uninteresting commercially, since the goal is to identify chatter or the start of chatter for interrupting the machinery, regardless of it being intense or not.

The method of clustering elbow was used to illustrate the algorithms' results. A clustering elbow is a method to identify the most optimal number of clusters to a problem, based on WCSS (Within-Cluster Sum-of-Squares), a measure of the variance within each cluster that evaluates how well that number of clusters classify the data. This method is the most popular and simple to use, (Naghizadeh and Metaxas, 2020). Thus, the use of more complex and time-consuming methods were deemed unnecessary. Since the files were already classified for three different classes, the algorithm suggested three classes, but since we were interested only in classifying chatter as present or not, it was specified for classification to be binary.

The reason why semi-supervised clustering was used to reclassify the windowed files was because, once split, different splits could have different chatter presence as the chatter originally detected could be present in one section of the file and not on another. The data that originally did not have chatter could not have chatter after slitting.

2.5 Feature Extraction

Feature extraction is a step in Machine Learning where relevant information is extracted from raw data and transformed into mathematical information. By creating mathematical features such as mean, standard deviation, RMS, Kurtosis, distortion, amplitude, and others, valuable statistical measures are captured, providing insights into different aspects of the signals generated by rotating machines. These carefully selected features enable the model to learn and identify patterns associated with chatter presence or absence.

To train the models, a new mathematical feature data frame was built. The features utilized were: mean, standard deviation (STD), root mean square (RMS), Kurtosis, distortion, amplitude, square root amplitude (SRA), crest, impulse, margin, maximum value of acceleration (MA), maximum frequency (MF), second-highest frequency (SHF), peak values and cluster. The Pearson Correlation between features was calculated to identify features' relevance and the impact caused by the use of a clustering method, as, if the correlation between original chatter presence and cluster was too high, the time demanded for clustering would be deemed unnecessary for an online tool. Features definition can be seen at Tab. 1 and these features are based on (Ribeiro, 2018).

Since the cluster obtained presented low relation with the other, all numerical features were utilized for training the model for allowing higher accuracies, since, if feature selection was applied, accuracies would be lowered by the low Pearson Correlations. The low correlation with chatter presence indicates that there were numerous files that needed to be reclassified, and also reflects the considerable amount of "intermediate" chatter presence classification that was clustered into either "chatter" or "without chatter". This low correlation and higher accuracies after the implementation of clustering prove the need for this step.

2.6 Model Training

Creating a tool for online chatter identification is promising to maximize profits, as discussed in the Introduction of this article. However, for this tool to be viable and efficient, it must be reliable and fast. Therefore, the criteria used to evaluate the models were the speed of the models and their accuracy. The faster and more accurate the models, the better they were

Table 1. Features definitions

Feature	Definition
Mean	$\mu_x = \frac{1}{N}\sum_{i=0}^{N} x_i$
STD	$\sigma_x^2 = \frac{1}{N}\sum_{i=0}^{N}(x_i - \mu_x)^2$
RMS	$x_{rms} = \left(\frac{1}{N}\sum_{i=0}^{N} x_i^2\right)^{1/2}$
Kurtosis	$\kappa_x = \frac{1}{N}\sum_{i=0}^{N}\left(\frac{x_i-\mu_x}{\sigma_x}\right)^4$
Distortion	$thd = \frac{\sqrt[100]{\sum_{i=0}^{N}(x_i - x_{max}^2))}}{x_{max}}$
Amplitude	$A = x_{max} - x_{min}$
SRA	$x_{sra} = \left(\frac{1}{N}\sum_{i=0}^{N}\sqrt{\lvert x_i\rvert}\right)^2$
Crest	$x_{cf} = \frac{max(\lvert x_i\rvert)}{x_{rms}}$
Impulse	$x_{if} = \frac{max(\lvert x_i\rvert)}{\frac{1}{N}\sum_{i=0}^{N}\lvert x_i\rvert}$
Margin	$x_{mf} = \frac{max(\lvert x_i\rvert)}{x_{sra}}$
MA (FFT)	Highest Peak value
MF (FFT)	$X_{max} = max(X)$
SHF (FFT)	Second-highest Peak value

(1) Source: (Ribeiro, 2018).

assessed. An efficient and well-researched tool for automatic classification is Machine Learning classifier algorithms, which were chosen for this project. Six different models were tested and optimized to identify the model with the best accuracy and fastest computation time for live detection and interruption of machinery, as speed and trustworthiness are crucial. The running time and accuracy of each model were measured. The running time is considered best when it is at its lowest, and the accuracy was arbitrarily considered best between 97% and 99%, as higher accuracy may indicate overfitting and lower accuracy may indicate insufficient reliability.

A Machine Learning classification model is used to classify a group of data based on the training it has undergone. Each model mathematically learns to distinguish classes of data and utilizes its respective algorithms to classify new incoming information. The models and kernels tested in this study were as follows:

- SVM with linear and RBF kernels: SVM is a classification algorithm that separates data points by finding the optimal hyperplane with the largest margin between classes. It maps the data into a high-dimensional space and finds the best decision boundary to maximize the separation between classes. The linear and RBF (Radial Basis Function) kernels allow it to handle linear and non-linear decision boundaries, respectively.

- Decision Tree: Decision Tree is a supervised learning algorithm that creates a tree-like model, where each internal node represents a feature, each branch represents a decision based on that feature, and each leaf node represents a class label. It recursively splits data based on features to create decision rules for classification or regression tasks.

- Random Forest: Random Forest is an ensemble learning method that constructs multiple decision trees and combines their predictions. Each tree is trained on a random subset of the data with replacement, and the final prediction is determined by averaging or voting the predictions from all trees. It helps to reduce overfitting and improves the overall accuracy and robustness of the model.

- KNN: KNN is a non-parametric algorithm used for classification and regression. It assigns a data point's class or value based on the majority vote or average of its k nearest neighbors in the feature space. It measures distance (typically Euclidean) to determine similarity between data points and makes predictions based on the neighbors' labels or values.

- ANN: ANN is a Machine Learning model inspired by the human brain's neural structure. It consists of interconnected nodes or "neurons" organized in layers and is capable of learning complex patterns and relationships in the data.

Several techniques were implemented to enhance the generalization of the models. Cross-validation, for example, involves splitting the training data into training and validation sets to assess and improve the model's generalization ability.

It helps estimate the model's performance on unseen data and reduces the bias from the training set selection. Enhancing generalization is important to improve the predictions made by the models on new, unseen data during training.

Furthermore, hyperparameter tuning was also employed to optimize the models. Hyperparameter tuning is a technique that tests different combinations of hyperparameter values in a Machine Learning model to find the most optimized model architecture. Many different tools for automatic hyperparameter tuning exist as to facilitate and fasten the process (Yang and Shami, 2020). The chosen tool for hyperparameter tuning was GridSearchCV, and the optimization criterion selected was accuracy. Each model was optimized with different parameters.

3. RESULTS

3.1 KMeans

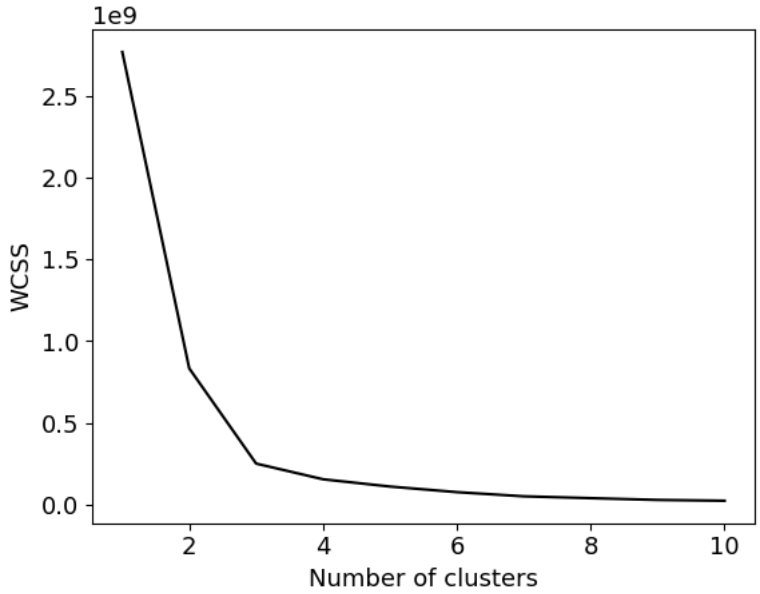

Figure 4. KMeans clustering elbow.

Table 2. Features' truncated Pearson Correlation.

Feature	Correlation with Chatter Presence	Correlation with Cluster
Mean	0.16	0.04
STD	-0.57	0.00
RMS	-0.51	0.00
Kurtosis	-0.42	0.03
Distortion	0.45	0.02
Amplitude	-0.53	0.01
SRA	-0.44	0.00
Crest	-0.60	0.02
Impulse	-0.60	0.02
Margin	-0.61	0.02
MA (FFT)	-0.51	0.01
MF (FFT)	0.00	-0.84
SHF (FFT)	0.00	-0.80
FFT	-0.38	-0.03
Cluster	0.07	–

(1) Source: authors.

A semi-supervised clustering model was run using KMeans to classify the files into two clusters. Using these clusters as target, the model type, its time taken for prediction and the measured accuracy are displayed in Tab. 3 for all models tested. The number of clusters chosen was only two, despite the elbow method in Fig. 4 indicating that three clusters was an appropriate number. This is because the tool in development is only supposed to classify the presence of chatter as

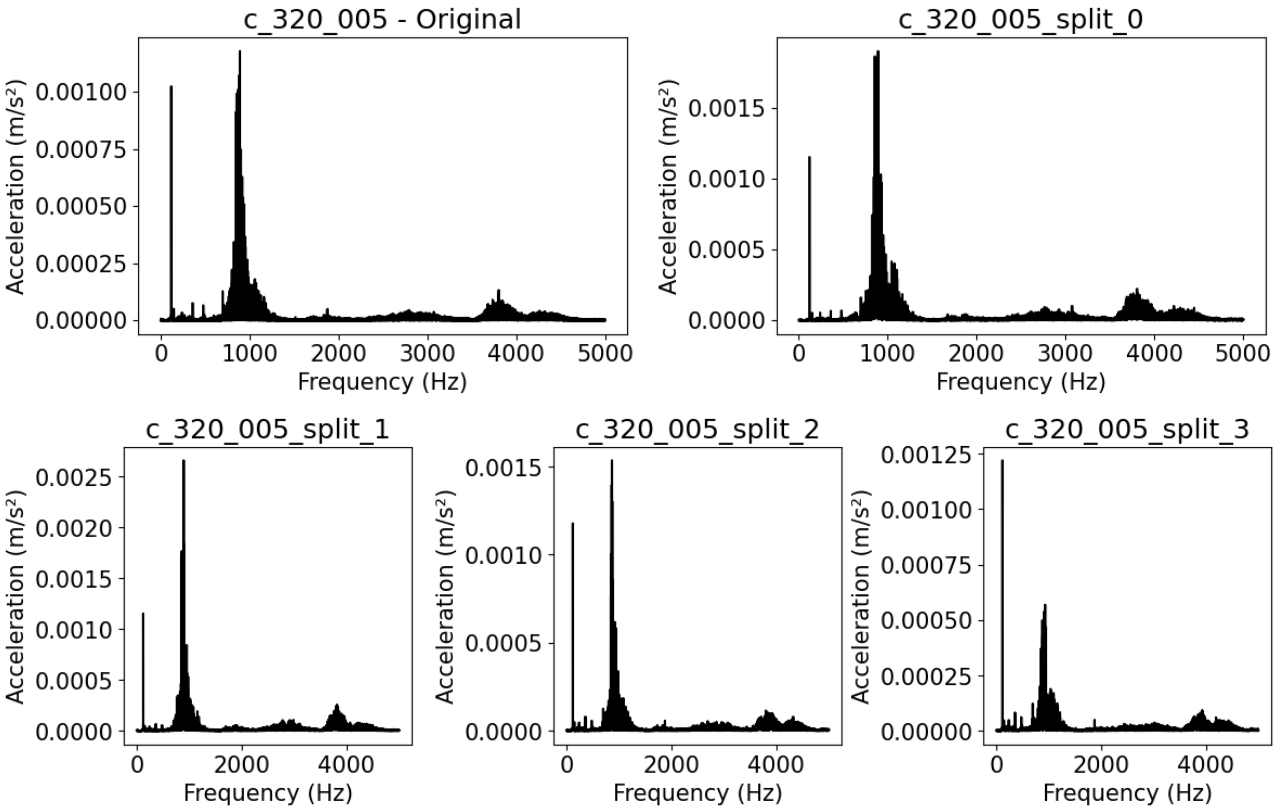

Figure 5. Fast Fourier Transformation (FFT) and Frequency for identifying chatter in different splits.

present or not. The detected chatters' intensity was not of interest, only its presence is relevant as the determining factor to interrupting the machinery.

Figure 2 shows how Windowing was performed: a file was split into four different files with equal lenghts. This resulted in the classification of the original files not being reliable for the split files. The chatter detected in the original files may be present in one split and not the other ones, as discussed, which can be seen in Fig. 5 that shows the acceleration and frequency for different split files that were windowed as shown in Fig. 2. This data was obtained with the use of Fast Fourier Transformation (FFT) on the original data. Semi-supervised clustering resulted in higher accuracies and more trustworthy results. Accuracies previous to the implementation of semi-supervised learning were around 70% to 80%.

As we can observe from the Pearson Correlation (Tab. 2), the clusters and the original classification were significantly distinct from each other. This demonstrates the need for reclassification and justifies the implementation of clustering. The good accuracy results obtained, as presented in Tab. 3, demonstrate the effectiveness achieved with the support of proper clustering.

3.2 Model Chosen

These data were obtained using all the mathematical attributes studied: mean, standard deviation, quadratic mean, Kurtosis, distortion, amplitude, square root of the amplitude, impulse, margin, maximum frequencies, second-highest frequency, and average frequency. Furthermore, we intend to test different combinations of these features and evaluate the performance-accuracy relationship to determine the fastest option to implement for real-time chatter detection without significant loss in accuracy. The total cross-validation execution time for all models took 0.8557 seconds.

The accuracy used for reference was the test prediction accuracy and the time measured used was the time necessary for making the predictions, which did not include the time necessary for fitting the model. The results can be seen at Tab. 3.

Since clustering and cross-validation didn't add up much time to the programs' execution, both have proven to be viable for live implementation without adding much delay to the detection of chatter since both have increased accuracy considerably.

Decision Tree with cross-validation presented the best relation between speed and accuracy, therefore is the model best fit for a live chatter detection system since one needs to be both fast and reliable to be economically and commercially viable. SVM with RBF kernel and GridSearchCV has also presented satisfactory results, although the accuracy is lower than Decision Trees. Random Forest with cross-validation showed good accuracy, but slower computational speed.

Table 3. Models' accuracy and time required for prediction.

Model	**Accuracy (%)**	**Time required (s)**
With cross-validation.		
SVM (Kernel linear)	98.63	0.0250
SVM (Kernel RBF)	94.52	0.0260
Random Forest	98.63	0.7336
Decision Tree	98.63	0.0180
KNN	91.78	0.0520
ANN	91.18	2.4906
With GridSearchCV.		
SVM (Kernel linear)	100	0.0010
SVM (Kernel RBF)	97.54	0.0010
Random Forest	99.18	0.0115
Decision Tree	99.18	0.0054
KNN	95.90	0.0157
ANN	82.35	–

(1) Source: authors.

Furthermore Decision Tree with cross-validation ranked as the best model for this project.

The worst models based on the same criteria are ANN, Decision Tree with GridSearchCV, Random Forest with GridSearchCV, and SVM with linear kernel and GridSearchCV since they showed accuracy outside the stablished limits of 97% to 99%. ANN also showed very irregular results each time the model was ran, showing a broad and unreliable range of accuracy and prediction time. ANN and KNN didn't reach the criteria with neither GridSearchCV nor cross-validation. Random Forest with GridSearchCV, SVM with linear kernel and GridSearchCV, and Decision Tree with GridSearchCV have all shown overfitting and therefore didn't meet the requirements.

4. CONCLUSION

The use of Machine Learning for chatter detection has proven to be efficient and yielded good accuracies. Thus, the feasibility of implementing Machine Learning, particularly the Decision Tree with cross-validation, has been established for an online chatter detection tool. These results are promising for this projects success.

The models that did not meet the established criteria were: SVM (RBF kernel and cross-validation), SVM (linear kernel and GridSearchCV), KNN (with both GridSearchCV and cross-validation), ANN (with both GridSearchCV and cross-validation), Random Forest with GridSearchCV, and Decision Tree with GridSearchCV.

The models that showed the best performance were: Decision Tree with cross-validation, SVM with linear kernel and cross-validation, SVM with RBF kernel and GridSearchCV and Random Forest with cross-validation. The model chosen as most appropriate for this project was Decision Tree with cross-validation.

Further feature selection testing is intended. Different clustering techniques or configuration may also be tested. Most prediction times were low, thus Machine Learning shows potential on the online detection of machining defects. The features used on this project, paired with semi-supervised clustering, resulted in great accuracies and reliable models.

5. ACKNOWLEDGEMENTS

We would like to thank everyone involved in the development and publishing of (Khasawneh *et al.*, 2019) dataset. As well as all the authors cited, whose work contributed greatly for this project. The authors are also thankful for the institutional support of PRPPG/UFPR through funding under Research Call 04/2023.

This work was funded by CAPES by means of a master CAPES-DS grant and by the INSTITUTIONAL PROGRAM OF SCIENTIFIC INITIATION SCHOLARSHIPS - UFPR.

6. REFERENCES

Ahmed, M., Seraj, R. and Islam, S.M.S., 2020. "The k-means algorithm: A comprehensive survey and performance evaluation". *Electronics*, Vol. 9, No. 8, p. 1295.

Khasawneh, F., Otto, A. and Yesilli, M., 2019. "Turning dataset for chatter diagnosis using machine learning". *Mendeley Data*, Vol. 1.

Naghizadeh, A. and Metaxas, D.N., 2020. "Condensed silhouette: An optimized filtering process for cluster selection in k-means". *Procedia Computer Science*, Vol. 176, pp. 205–214.

Oleaga, I., Pardo, C., Zulaika, J.J. and Bustillo, A., 2018. "A machine-learning based solution for chatter prediction in heavy-duty milling machines". *Measurement*, Vol. 128, pp. 34–44.

Paul, A.R., Biswas, S. and Mukherjee, M., 2022. "Conceptualisation of a novel technique to incorporate artificial intelligence in preventive and predictive maintenance in tandem". *Materials Today: Proceedings*, Vol. 66, pp. 3814–3821.

Quintana, G. and Ciurana, J., 2011. "Chatter in machining processes: A review". *International Journal of Machine Tools and Manufacture*, Vol. 51, No. 5, pp. 363–376.

Ribeiro, F.M.L., 2018. "Similarity-based methods for machine diagnosis". Ph.D. thesis, Graduate Program in Electrical Engineering, Federal University of Rio de Janeiro, Rio de Janeiro, Brasil.

Sestito, G.S., Venter, G.S., Ribeiro, K.S.B., Rodrigues, A.R. and da Silva, M.M., 2022. "In-process chatter detection in micro-milling using acoustic emission via machine learning classifiers". *The International Journal of Advanced Manufacturing Technology*, Vol. 120, No. 11-12, pp. 7293–7303.

Wen, Q., Sun, L., Yang, F., Song, X., Gao, J., Wang, X. and Xu, H., 2021. "Time series data augmentation for deep learning: A survey". *arXiv preprint arXiv:2002.12478*, pp. 4653–4660.

Yang, L. and Shami, A., 2020. "On hyperparameter optimization of machine learning algorithms: Theory and practice". *Neurocomputing*, Vol. 415, pp. 295–316.

7. RESPONSABILITY NOTICE

The authors are the only responsible for the printed material included in this paper.