Journal of Machine Engineering, 2025, Vol. 25
ISSN 1895-7595 (Print) ISSN 2391-8071 (Online)

Received*: 11 February 2025* / Accepted: *07 April 2025* / Published online: *23 May*

machine learning,
five-axis machine tool,
volumetric errors,
R-test

Min ZENG[1*], J.R.R. MAYER[1],
Miao FENG[2], Elie BITAR-NEHME[1],
Xuan Truong DUONG[3]

COMPARISON OF TWO MACHINE LEARNING MODELS FOR PREDICTING VOLUMETRIC ERRORS FROM ON-THE-FLY R-TEST TYPE DEVICE DATA AND VIRTUAL END POINT CONSTRAINTS

On-the-fly virtual end-point constraints consists in moving all five axes of the machine tool while nominally maintaining the coincidence of a sensing head centre point with a master ball centre attached to the workpiece table. The sensing head detects the deviations from the nominal coincidence as a 3D volumetric error vector. More than one ball can be so measured, and a fixed length ball bar is also measured for detecting isotropic scaling effects. Initial processing of data using the SAMBA (scale and master ball artefact) method eliminates setup errors and provides estimates of inter- and intra-axis errors as well as volumetric error vectors. Two ML models are trained and compared, Neural Network (NN) and eXtreme Gradient Boosting (XGBoost), to find the most suitable model and the required amount of training data to predict volumetric errors of a five-axis machine tool with wCBXfZY(S)t topology based on axis commands. The results show that NN marginally outperforms XGBoost and a kinematic model with ratios of prediction error over volumetric error norms of 0.12, 0.13 and 0.14, respectively.

1. INTRODUCTION

To avoid complexity and the requirement for expert knowledge, a growing number of researchers are applying machine learning (ML) methods to predict volumetric errors (VEs). Nguyen et al. [1, 2] used a dataset of 150 samples, collected by Li et al. [3] using an API Radian-20 laser tracker along with an active target (AT), consisting of *X*, *Y*, and *Z* axis positions, and *A* and *C* rotation angles as inputs with the corresponding VEs used as outputs for a large five-axis gantry tilting head machine tool. In [1], Nguyen et al. trained a three-layer artificial neural network using 70% of the dataset (105 samples), and validated the

[1] Department of Mechanical Engineering, Polytechnique Montréal, Canada
[2] Department of Computer Science and Operations Research, Université de Montréal, Canada
[3] Department of Mechanical Engineering, Dawson College, Canada
[*] E-mail: min.zeng@polymtl.ca
https://doi.org/10.36897/jme/203805

performance of the model using the remaining 30% of the dataset (45 samples) resulting in a 0.95 coefficient of correlation of the validation set. Subsequently in [2], the authors trained linear, Support Vector Machine (SVM), Extreme Gradient Boosting (XGB), Artificial Neural Networks (ANNs) with Single, MultiOutput and Chained MultiOutput regression using 120 training data and tested their performances using 30 testing data. They found that the SVR with Single regression model had the best VEs predictive performance with an *RMSE* of 0.032 mm, which is a 27% improvement over the polynomial technique. Wan et al. [4] used a laser tracker to measure the nominal positions of sixty and thirty randomly selected fiducial points, respectively. They then trained a Gaussian process regression based (GPR-based) model using nominal positions as input, and VEs as outputs. The performance of the model was then tested with a new dataset of twenty randomly selected positions. The mean VE is 0.16 mm, and the mean prediction error is 0.02 mm with all prediction errors being below 0.05 mm. Guo et al. [5] used a laser tracker to collect VEs from a two-turntable five-axis CNC machine tool as output, position information, temperature, and spindle motor current as input. The researchers trained ACO-BPN (ant colony algorithm-based back propagation neural network), GA-BPN (Genetic Algorithm with Back Propagation Network), PPR (Project Pursuit Regression) and a fusion model that combined the aforementioned three models. After comparing their performances, the residual errors of the fusion model were reported to be consistently smaller than 2 μm, indicating its higher precision in predicting and compensating VEs compared to those three models. Ngoc et al. used Capball, an R-test type device with five non-contact precision capacitive proximity sensors, to measure raw translational volumetric deviations over 132 hours (528 cycles) to train a Stacked Long Short Term Memory (LSTM) model for predicting geometric error [6]. They concluded that the Stacked LSTM model using the Adam optimizer showed superior performance, predicting up to 93% of the main geometric error (EXX1) when compared to other optimizers. Additionally, they used the model to predict thermally induced VEs [7]. To test the model's performance, they measured raw volumetric deviations over 40 hours (160 cycles). The best prediction error of the SLSTM model for the long process was 2.2 μm of *RMSE* (98.8% fit to experimental measurements), while in the short test process it was 1.8 μm of *RMSE* (91.6% fit to experimental measurements). The worst case prediction error by the model was 7.5 μm of *RMSE* (fit of 69.2%). Liu et al. [8] used the laser interferometer Renishaw XL80 to measure the positioning errors of the *X*-, *Y*- and *Z*-axes. The thermal errors were then decomposed from these measurements. Temperatures, angular and translational errors of the spindle system, feed rate and rotational speed as inputs, to predict thermal errors with a Multivariate Linear Regression Analysis-based model. The study achieved a maximum fitting accuracy of 95% using the polynomial model to fit the positioning error data of the linear axes.

The dataset in the abovementioned studies, were from previous references or based on a limit amount. The main contribution of this paper is (1) to collect a large amount of data in a short time using on-the-fly virtual end-point constraints method, (2) to propose a process of matching programming point and Capball reading and (3) to find the most suitable ML model and the required amount of training data. Experimental procedure is explained in Section 2, structure and hyperparameters of the used machine learning model is explained in Section 3. Followed by results explained in Section 4, and finally draws the conclusion in Section 5.

2. EXPERIMENTAL PROCEDURE

2. 1. EXPERIMENTAL SETUP

Translational volumetric error vectors (VEs) are collected using an in-house developed device named Capball during a SAMBA (scale and master ball artefact) test conducted on a Mitsui-Seiki HU40-T five-axis horizontal machine tool with a wCBXfZY(S)t topology (Fig. 1), which is similar as the one in [7]. A fixed length, magnetically and kinematically held, double ball bar, and a RUMBA (reconfigurable uncalibrated master ball artefacts) of four ceramic balls Ø12.7 with different nominal lengths of 102, 128, 153 and 172 mm are installed on the machine pallet. The Capball [9], which is an R-test type device with five non-contact capacitive sensors, is mounted in the tool holder.

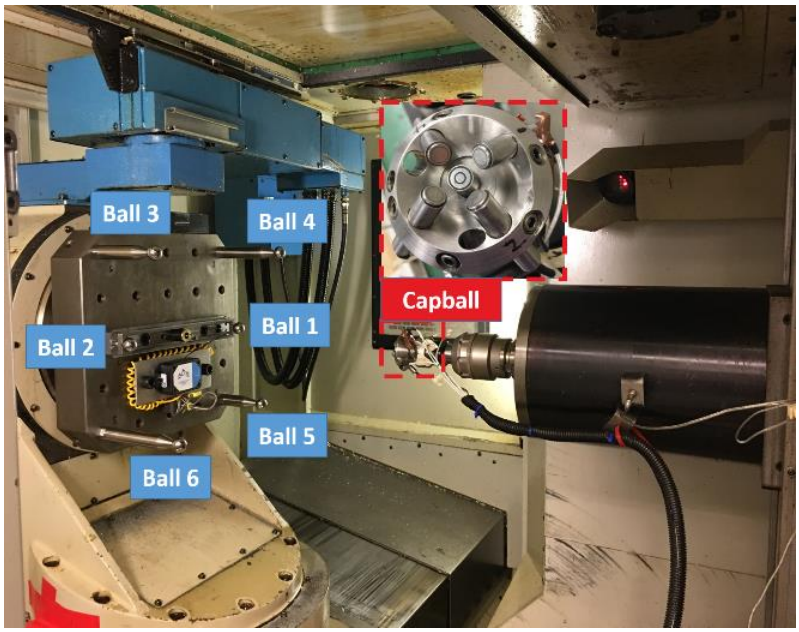

Fig. 1. Scale and master balls artefact (SAMBA) ready for collecting data by the Capball on a wCBXfZY(S)t machine; shown for $b = 0°$, $c = 0°$

During the experiment, the on-the-fly [10] virtual end-point constraints consists in moving all five axes of the machine tool while nominally keeping the center point of the sensing head coincide with the centre of a master ball, attached to the workpiece table, with the B-axis moving in the range from –90 to +90 in 0.2-degree increments (Fig. 2), and the *C*-axis moving in the range –360 to +360 in 0.1 or 0.2 degree increments (Fig. 2). This process was named chase-the-ball originally in [11]. To avoid collision, at some angles, the Capball moves away from the chased master ball and the Capball approaches again to chase the master ball until the ball moves to the next indexation. In addition, to detect scaling effects, a fixed length ball bar is also measured. The five sensor voltage readings from the Capball are recorded (Fig. 3 – Fig. 6), converted to mm and combined to produce the translational VEs.

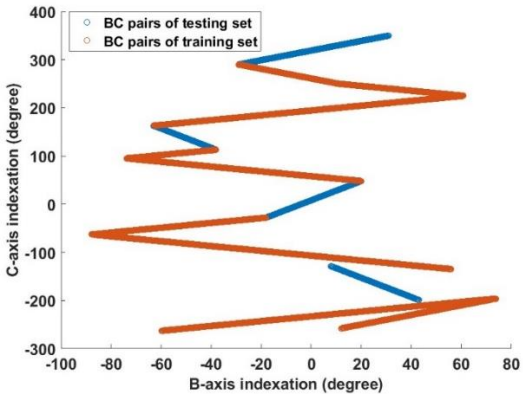

Fig. 2. (b_i, c_i) pairs of training set and testing set

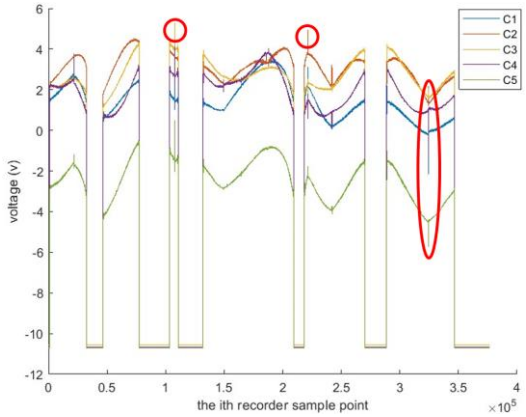

Fig. 3. Raw data of Ball 3

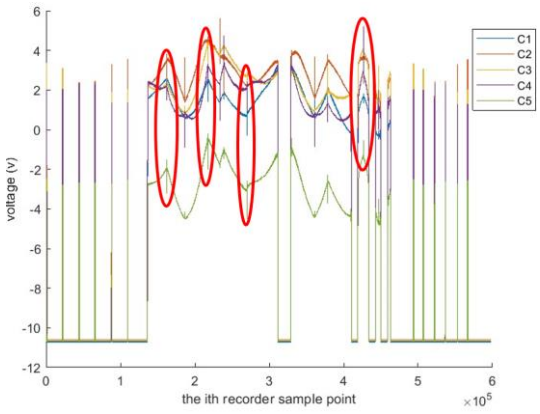

Fig. 4. Raw data of Ball 4

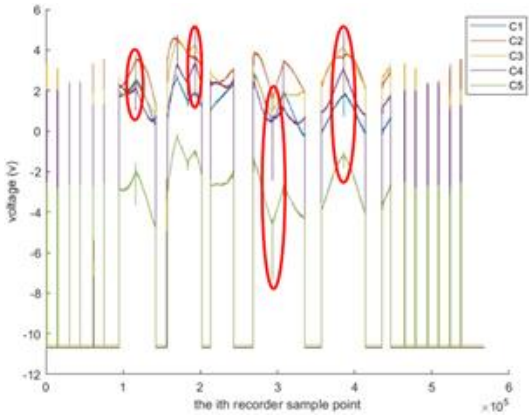

Fig. 5. Raw data of Ball 5

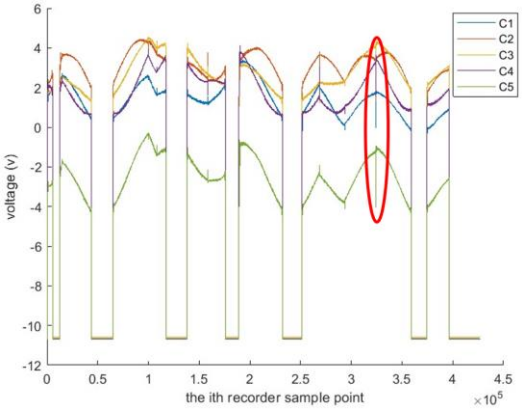

Fig. 6. Raw data of Ball 6

2.3. DATA PROCESSING

In this study, the sudden VEs spikes, showed in the red ovals in Fig. 3 – Fig. 6, are removed as we will make no attempt at predicting them.

On-the-fly measurement is performed in order to reduce test time. To avoid collisions, the entire measurement process of each master ball is divided into several segments. At the beginning and end of each phase, the machine dwells for 0.3 s, which causes the data to flat line to easily associate the data with the machine motion and so automate the data processing. To match each programming point with the corresponding recorded voltage, a process is developed to calculate the number of sample points between each programming position.

To realize a smooth movement between each programming point, the whole experiment is conducted based on a programmed G-code using G93 command. According to the *F* value in the G-code, the time used from one programming point to the next is calculated with Equation (1),

$$t_i = \frac{1}{F_{i+1}} \cdot 60 \ \left(i = 1, 2, 3, \ldots, (k-1)\right) \tag{1}$$

where k represents the total number of programming points; F_{i+1} is the *F* value to complete the movement from the ith programming point to the $(i+1)th$ programming point in the unit of min/motion unit; t_i is the corresponding time of the movement from the ith programming position to the $(i+1)th$.

Then, the theoretical number of data samples collected by the Capball during each movement is obtained by Equation (2)

$$np_{cal_i} = t_i \cdot 1000 \ \left(i = 1, 2, 3, \ldots, (k-1)\right) \tag{2}$$

where $np_{cal_i}$is each calculated number of sample points from the ith programming point to the $(i+1)$th programming point, with 1000 being the used sampling rate in Hz.

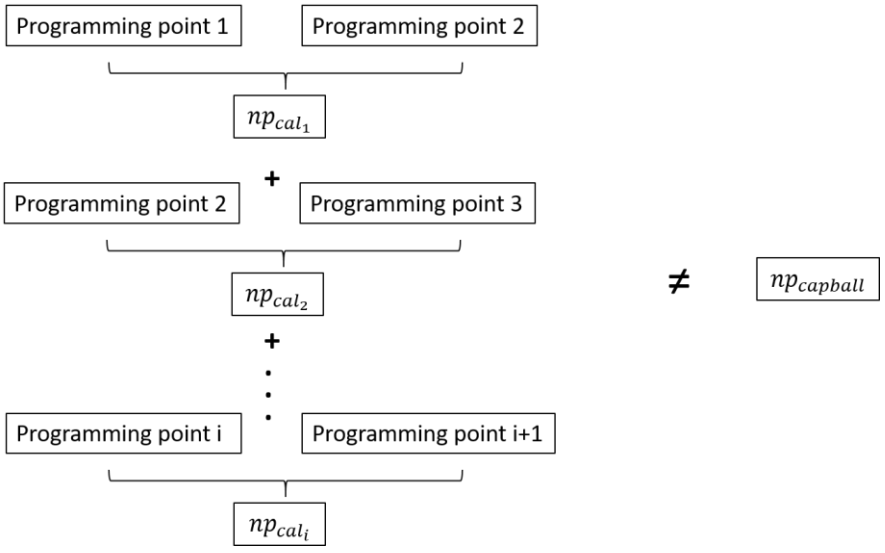

Fig. 7. Difference between calculated number of sample points and the actual number of sample points

However, there is a difference between the total calculated number of sample points and the actual number of sample points (Fig. 7), a general process consisting of four steps is developed to adjust the difference. The pseudocode is listed below:

Step1: Calculate the ratio r using Equation (3) of the actual number of sample points and the calculated one, and then, rounding the result to an integer based on the "$roundn$" function (Equation (3)) in Matlab®.

$$r = roundn\left(\frac{np_{capball}}{\sum_{i=1}^{k-1} np_{cal_i}}, 0\right) \left(i = 1, 2, 3, \dots, (k-1)\right) \tag{3}$$

where $np_{capball}$ is the actual number of sample points, and np_{cal_i} is the calculated number of sample points between each programming point.

Step 2: Obtain the difference (δ) between actual number of sample points and the calculated one multiply by the ratio calculated in Equation (4).

$$\delta = np_{capball} - \sum_{i=1}^{k-1} np_{cal_i} \cdot r \left(i = 1, 2, 3, \dots, (k-1)\right) \tag{4}$$

Step 3: Calculate the quotient (Equation (5)) and remainder (Equation (6).

$$quotient = floor\left(\frac{\delta}{k}\right) \tag{5}$$

$$remainder = mod(\delta, k) \tag{6}$$

Step 4: Adjust the calculated number of sample points based on Equation (7) and Equation (8).

$$np_{adjust_i} = np_{cal_i} \cdot r + quotient \left(i = 1, 2, 3, \dots, (k-1)\right) \tag{7}$$

$$np_{adjust}(1{:}remainder, 1) = \left(np_{adjust}(1{:}remainder, 1)\right) + 1 \tag{8}$$

If v_j is the recorded voltage correspond to the ith programming point, $v_{j+np_{adjust_1}} will\ be$ the recorded voltage correspond to the $(i+1)th$ programming point.

Finally, the program developed in [9] is used to convert voltages (Fig. 3 - Fig. 6) after removing the outliers (spikes highlighted in red ovals in Fig. 3 - Fig. 6, with abnormal voltage changes greater than 5%) to the raw VE of each programming point. The machine kinematic error model and measurement model [12] are then applied to compute the translational volumetric error vectors and the kinematic error parameters.

The approximate ratio of 7:1 is used to split the data into training set and testing set. To ensure the (b_i, c_i) pairs of the testing set are different from those in the training set, we select 8346 samples (from Ball 3, 5, and 6) plus four samples of scale bar used for training and 1170 samples (from Ball 4) for testing. The (b_i, c_i) pairs of testing set in orange are different from (b_i, c_i) pairs of training set in blue (Fig. 2).

3. STRUCTURE AND HYPERPARAMETERS OF THE USED MACHINE LEARNING MODEL

3. 1. NEURAL NETWORK

A three-layer Neural Network (NN) is used (Fig. 8). In the first hidden layer, 512 neurons, with the activation of 'relu' is implemented. In the second hidden layer, eight neurons and "relu" is applied. As a regression problem, the last layer uses 'linear' activation, and three neurons are used to output volumetric errors in the *X*, *Y*, and *Z* directions. We use 1000 epochs and batch size is eight. To train the NN model efficiently, a reduced learning rate strategy and early stop are used. When the loss (value of "mean_squared_error") does not change for 30 epochs, the learning rate will be reduced until the value reaches to 1e-6, and after 50 epochs' non changeable loss, the training process will stop. All the parameters (Table 1) mentioned is determined by "Talos", the simplest and most powerful available method for hyperparameter optimization with Keras. The details of each hyperparameter can be found in [13].

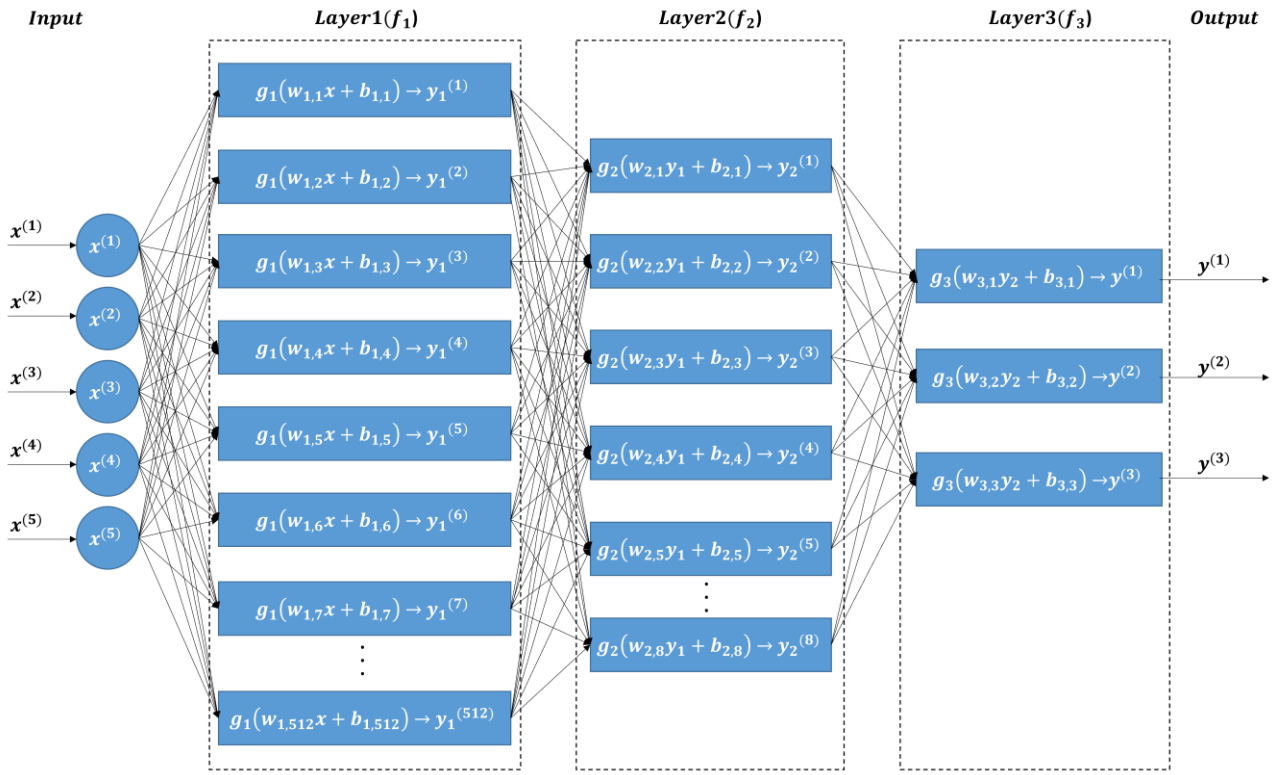

Fig. 8 Structure of Neural Network (adapted from [14])

Table 1. Hyper-parameters of NN

Number of layers	Three		Number of neurons	Activation
		First hidden layer	512	"relu"
		Second hidden layer	8	"relu"
		The third layer	3	"linear"
batch_size	8			
epochs	1000			
optimizer	"Adam"			
learning_rate	1e-4			
minmum learning_rate	1e-6			
loss	"mean_squared_error"			

3. 2. EXTREME GRADIENT BOOSTING

A Gridsearch Cross-Validation (CV) and Bayes search CV are applied and compared to determine the hyperparameters (Table 2) in eXtreme Gradient Boost (XGBoost). The details of each hyperparameter can be found in [15].

Table 2. Hyper-parameter in XGBoost

max_depth	5
learning_rate	0.12
objective	'reg:squarederror'
n_estimators	1250
booster	'gbtree'
colsample_bytree	0.8
subsample	0.8

3. 3. MODEL PERFORMANCE ASSESSMENT CRITERIA

To assess the performance of the proposed models, four metrics are employed: root mean square error (*RMSE*) [16], mean absolute error (*MAE*) [16], *fitting* [7] and prediction error norm ratio with respect to the norm of the VEs ($PENR_{VEs}$). They are calculated using Equations (9), (10), (11) and (12), where n is the number of observations, $\hat{y}$ are the predicted values, y are the true target values, and y_{max} and y_{min} are the maximum and minimum values of the true target value respectively.

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(\hat{y}_i - y_i)^2}{n}} \tag{9}$$

$$MAE = \frac{\sum_{i=1}^{n}|\hat{y}_i - y_i|}{n} \tag{10}$$

$$fitting(\%) = \left(1 - \frac{RMSE}{y_{max} - y_{min}}\right) \times 100 \tag{11}$$

$$PENR_{VEs} = \frac{\|\boldsymbol{e}\|}{\|VEs\|} \tag{12}$$

where $\|\boldsymbol{e}\|$ and $\|VEs\|$ are the norm of prediction error and norm of VEs, respectively.

4. RESULTS AND DISCUSSIONS

4.1. COMPARISON OF PREDICTIVE PERFORMANCE USING DIFFERENT AMOUNT OF TRAINING DATA BEFORE CORRECTION FOR EX(0B)C

The kinematic model (green line with circles) consistently achieves the lowest *RMSE* (Fig. 9), *MAE* (Fig. 10), and the highest *fitting* (Fig. 11) in the *X* and *Y* directions, demonstrating its stability across varying training sizes in those two directions. However, a noticeable performance improvement in the NN model (orange line with triangles) is observed in the *Z* direction when the training samples exceed 265, where it outperforms both kinematic model and XGBoost (blue line with squares). XGBoost has the highest variability and always performs the worst.

In the *X* direction, increasing the training size tends to reduce the *RMSE*, *MAE* and increase *fitting* for all three models initially. As the training size continues to increase, *RMSE*, *MAE* and *fitting* of kinematic model and NN converge to relatively stable values, while those of XGBoost remain volatile.

In the *Y* direction, training with more data does not improve the performance of all three models.

In the *Z* direction, as the size of the training dataset increases, *RMSE*, *MAE* of NN decrease and *fitting* of NN increases and finally outperforms the kinematic model and XGBoost.

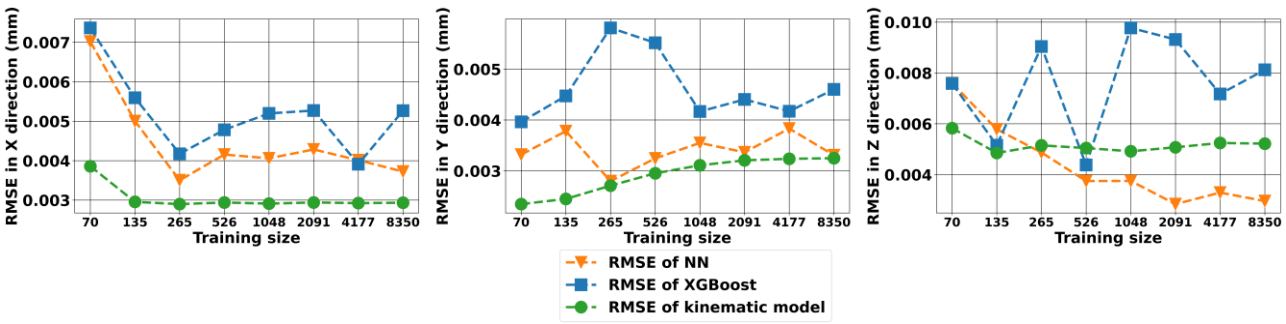

Fig. 9. *RMSE* of NN, XGBoost and kinematic model trained with different training size

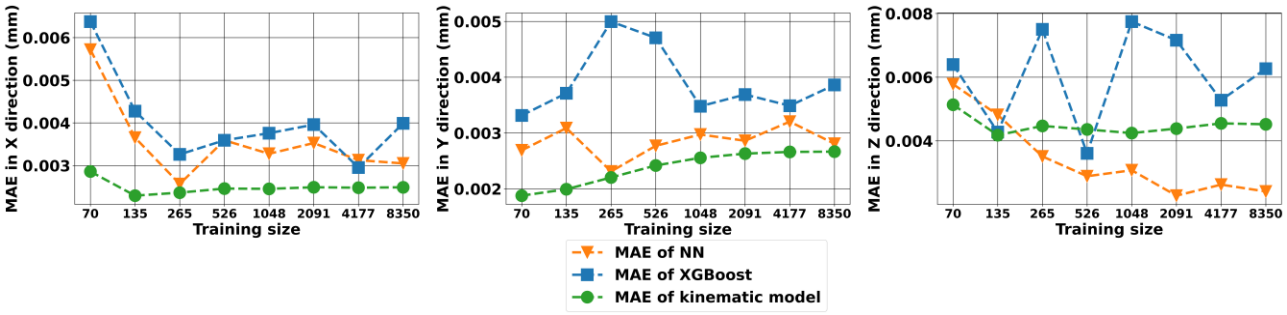

Fig. 10. *MAE* of NN, XGBoost and kinematic model trained with different training size

For an overall assessment of the performance of each model, average prediction error norm ratios with respect to the norm of the VEs ($\overline{PENR_{VEs}}$) are calculated and plotted in Fig. 12. The $\overline{PENR_{VEs}}$ of the kinematic model (green line with circles) and NN (orange line with triangles) decreases initially when training data increases from 70 to 135 and 265 respectively, after then they fluctuate. For NN, when training data increases to 8350, it reaches its best performance with the lowest $\overline{PENR_{VEs}}$ of 0.05. While the $\overline{PENR_{VEs}}$ of XGBoost (blue line with squares) has the largest variability and is always the largest. With respect to $\overline{PENR_{VEs}}$, NN outperforms the kinematic model and XGBoost from a training dataset size of 2091.

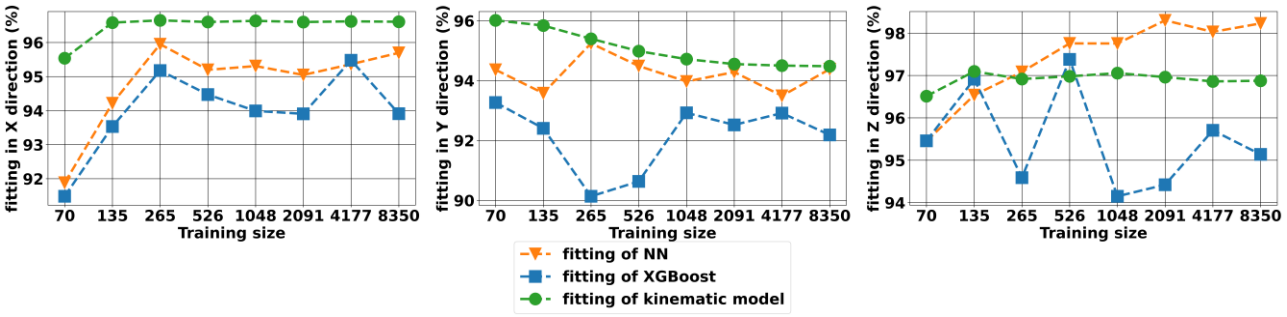

Fig. 11. *Fitting* of NN, XGBoost and kinematic model trained with different training size

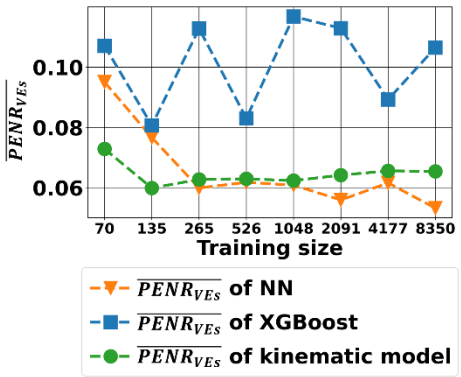

Fig. 12. Average prediction error norm ratio with respect to the norm of the VEs of NN, XGBoost and kinematic model trained with different training size

4.2. COMPARISON OF PREDICTIVE PERFORMANCE USING DIFFERENT AMOUNT OF TRAINING DATA AFTER CORRECTION FOR EX(0B)C

The tested machine tool has a large and dominant EX(0B)C inter-axis geometric error, which may have a significant impact on the performance of the various model performance. In this section, the models are trained using data that have been corrected by removing the volumetric effects of EX(0B)C. Fig. 13, Fig. 14 and Fig. 15 illustrate *RMSE*, *MAE* and *fitting*, respectively, of the kinematic, NN and XGBoost models based on different amount of training data after correction for EX(0B)C.

In the *X* direction, the kinematic model (green line with circles) performs the best with the overall lowest *RMSE*, *MAE* and the highest *fitting*. In this direction, the performance of all three models improves rapidly initially, and perform differently afterwards. For the kinematic model, it remains relatively stable with minor variations. The performance of XGBoost continuous to improve with the increase of training size, but fluctuates slightly. NN worsens (higher *RMSE* and *MAE*, lower *fitting*) until training size reaches 4177, after which it improves again.

In the *Y* direction, the performances of all three models show a slight decreasing trend with the increase of the training data, whereas XGBoost performs the worst.

Kinematic model performs the worst in the *Z* direction with the overall largest *RMSE*, *MAE* and the smallest *fitting*. There is a sharp improvement of NN initially, and it fluctuates afterwards.

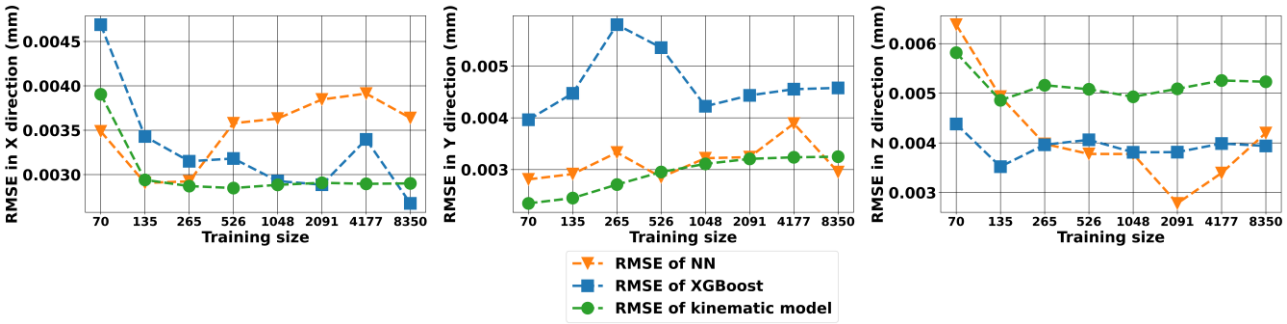

Fig. 13. *RMSE* of kinematic model, NN and XGBoost based on different training size after correction for EX(0B)C

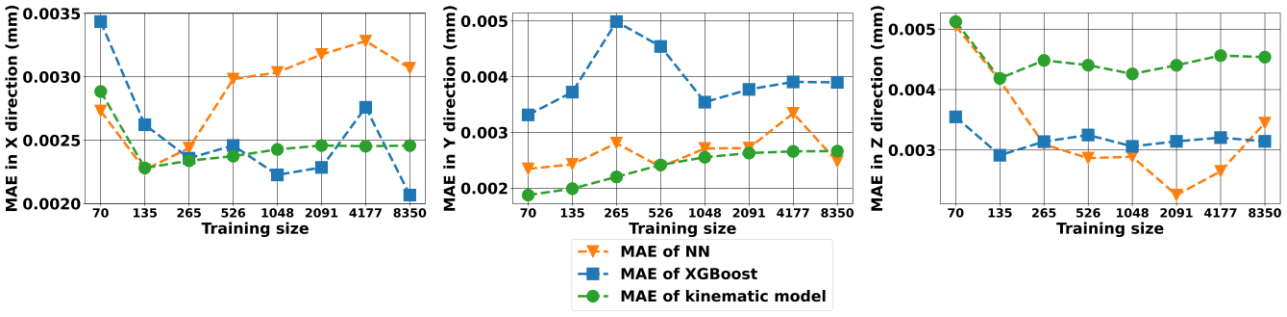

Fig. 14. *MAE* of kinematic model, NN and XGBoost based on different training size after correction for EX(0B)C

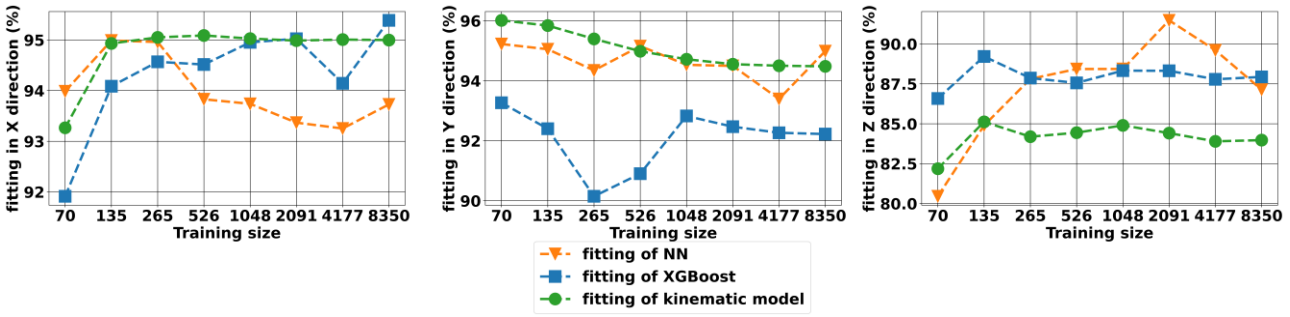

Fig. 15. *Fitting* of kinematic model, NN and XGBoost based on different training size after correction for EX(0B)C

Since no model performs always better than the others in all directions, average prediction error norm ratio with respect to the norm of the VEs ($PENR_{VEs}$) is calculated and plotted in Fig. 16 to provide with an overall performance of each model. NN is sensitive to the number of training dataset, and has an overall drop tendency till the training size decrease to 2091. While, training size does not have a big impact on the performance of the kinematic model. When training size becomes larger than 265, the performance of NN starts to outperform both the kinematic model and XGBoost. When the training data increases to 2091, the average prediction error norm ratio of NN is the smallest.

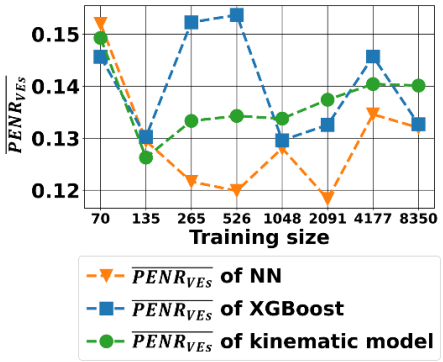

Fig. 16. Average prediction error norm ratio with respect to the norm of the VEs of NN, XGBoost and kinematic model based on different training size after correction for EX(0B)C

According to the results based on assessment criteria: *RMSE* (Fig. 17), *MAE* (Fig. 18), and *fitting* (Fig. 19), in the *Z* direction, NN trained with 2091 samples (three master balls with 1916 (b_i, c_i) pairs plus one scale bar) performs the best with the lowest *RMSE*, *MAE* and the highest f*itting*, while being the worst in the *X* direction. XGBoost performs the worst in the *Y* direction. However, according to the *fitting* results, the performance of all three models are very close to each other, especially in the *X* and *Y* directions.

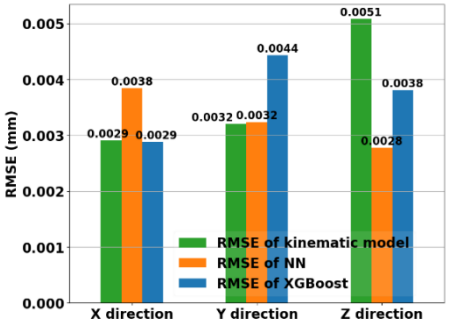

Fig. 17. *RMSE* of NN, XGBoost and kinematic model using 2091 training samples (three master balls with 1916 (b_i, c_i) pairs plus one scale bar) after correction for EX(0B)C

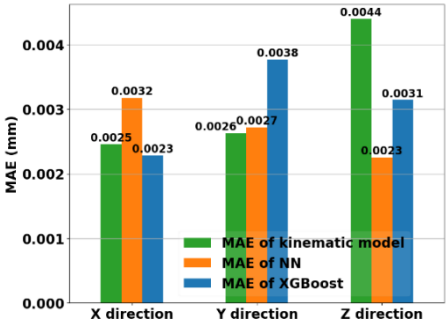

Fig. 18. *MAE* of NN, XGBoost and kinematic model using 2091 training samples (three master balls with 1916 (b_i, c_i) pairs plus one scale bar) after correction for EX(0B)C

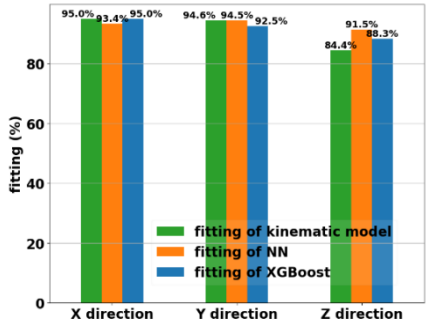

Fig. 19. *Fitting* of NN, XGBoost and kinematic model using 2091 training samples (three master balls with 1916 (b_i, c_i) pairs plus one scale bar) after correction for EX(0B)C

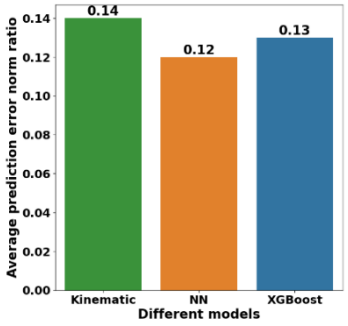

Fig. 20. Average prediction error norm ratio of NN, XGBoost and kinematic model using 2091 training samples (three master balls with 1916 (b_i, c_i) pairs plus one scale bar) after correction for EX(0B)C

In addition, we plotted average prediction error norm ratio of NN, XGBoost and kinematic model (Fig. 20). The result shows that, when 2091 training samples are used, NN outperforms XGBoost and the kinematic model with ratios of prediction error over volumetric error norms of 0.12, 0.13 and 0.14, respectively.

5. CONCLUSIONS AND FUTURE WORK

5.1. CONCLUSIONS

In this study, we collect experimental data using an R-test type device with on-the-fly data gathering. Two ML models are trained and compared: Neural Network (NN) and eXtreme Gradient Boosting (XGBoost), to predict volumetric errors of a five-axis machine tool with wCBXfZY(S)t topology based on axis commands. The results show that by using 2091 training samples (three master balls with 1916 (b_i, c_i) pairs plus one scale bar), NN outperforms XGBoost and is comparable to a kinematic model (with the machine error parameters of EA(0Z)B, EC(0X)B, EX(0B)C, EA(0B)C, EB(0X)C, EB(0X)Z, EA(0Z)Y,

EC(0X)Y, EXX, EYY and EZZ) with ratios of prediction error norms over volumetric error norms of 0.12, 0.13 and 0.14, respectively. It is found that the number of training samples has an impact on the performance of both NN and XGBoost.

This study contributes to the general knowledge by demonstrating that ML models, particularly NN, can achieve comparable accuracy to physics-based models when provided with sufficient training data. It also reveals that the performance of NN is sensitive to the amount of training data, which provides insight for future model selection and planning of training size. For example, if it is difficult to collect large amounts of data due to time, cost or machine constraints, we could reduce the training samples to 526, as the ratio of prediction error over volumetric error norms only drops by about 1% compared to 2091 training samples. These findings suggest that the data-driven approach can serve as practical, adaptive alternatives to traditional error modelling methods, especially when machine error parameters are difficult to identify or vary over time.

5.2. FUTURE WORK

NN performs better than the kinematic model, but not significantly. In the next project, we will study more features other than axis commands such as axis motion sense and also enriching the path complexity to improve the performance of ML models.

Moreover, only two ML models (NN and XGBoost) are evaluated in this study, because of their capability of handling nonlinear data. In the future, we plan to apply the concept of "meta-learning" or "double-loop" to compare a broader range of machine learning models, including support vector machines (SVM), with the aim of finding better-performing models.

ACKNOWLEDGEMENTS

This research is funded by NSERC's Discovery grant RGPIN-2022-04092. The authors would like to thank technicians Guy Gironne and Vincent Mayer for their help with the experimental work.

REFERENCES

[1] NGUYEN V.-H., LE T.-T., 2022, *Developing Geometric Error Compensation Software for Five-Axis CNC Machine Tool on NC Program Based on Artificial Neural Network*, Advances in Asian Mechanism and Machine Science, Conference paper, 113, 541–548.

[2] NGUYEN V.-H., LE T.-T., TRUONG H.-S, DUONG H.T., LE M.V., 2023, *Predicting Volumetric Error Compensation for Five-Axis Machine Tool Using Machine Learning*, International Journal of Computer Integrated Manufacturing, 36, 1191–1218.

[3] LI Q., WANG W., ZHANG J., SHEN R., LI H., JIANG Z., 2019, *Measurement Method for Volumetric Error of Five-Axis Machine Tool Considering Measurement Point Distribution and Adaptive Identification Process*, International Journal of Machine Tools and Manufacture, 147, 103465.

[4] WAN A., SONG L., XU J., LIU S., CHEN K., 2018, *Calibration and Compensation of Machine Tool Volumetric Error Using a Laser Tracker*, International Journal of Machine Tools and Manufacture, 124, 126–133.
[5] GUO Q., XU R., MAO C., XU H., YANG J., 2014, *Application of Information Fusion to Volumetric Error Modeling of CNC Machine Tools*, The International Journal of Advanced Manufacturing Technology, 78, 439–447.
[6] VU NGOC H., MAYER J.R.R., BITAR-NEHME E., 2022, *Deep Learning LSTM for Predicting Thermally Induced Geometric Errors Using Rotary Axes Powers as Input Parameters*, CIRP Journal of Manufacturing Science and Technology, 37, 70–80.
[7] NGOC H.V., MAYER J.R.R., BITAR-NEHME E., 2023, *Deep Learning to Directly Predict Compensation Values of Thermally Induced Volumetric Errors*, Machines, 11, 496.
[8] LIU J., MA C., WANG S., 2020, *Data-Driven Thermally-Induced Error Compensation Method of High-Speed and Precision Five-Axis Machine Tools*, Mechanical Systems and Signal Processing, 138, 106538.
[9] BITAR-NEHME E., MAYER J.R.R., 2016, *Thermal Volumetric Effects Under Axes Cycling Using an Invar R-Test Device and Reference Length*, International Journal of Machine Tools and Manufacture, 105, 14–22.
[10] ZARGARBASHI S.H.H., MAYER J.R.R., 2009, *Single Setup Estimation of a Five-Axis Machine Tool Eight Link Errors by Programmed End Point Constraint and on the Fly Measurement with Capball Sensor*, International Journal of Machine Tools and Manufacture, 49, 759–766.
[11] BRINGMANN B., KNAPP W., 2006, *Model-Based Chase-The-Ball Calibration of A 5-Axes Machining Center*, CIRP Annals, 55, 531–534.
[12] MAYER J.R.R., 2012, *Five-Axis Machine Tool Calibration by Probing a Scale Enriched Reconfigurable Uncalibrated Master Balls Artefact*, CIRP Annals, 61, 515–518.
[13] YU T., ZHU H., 2020, *Hyper-Parameter Optimization: a Review of Algorithms and Applications*, arXiv preprint arXiv:2003.05689.
[14] BURKOV A., 2019, *The Hundred-Page Machine Learning Book*, Andriy Burkov Quebec City, QC, Canada, Vol. 1.
[15] KAVZOGLU T., TEKE A., 2022, *Advanced Hyperparameter Optimization for Improved Spatial Prediction of Shallow Landslides Using Extreme Gradient Boosting (Xgboost)*, Bulletin of Engineering Geology and the Environment, 81.
[16] CHICCO D., WARRENS M.J., JURMAN G., 2021, *The Coefficient of Determination R-Squared is More Informative than SMAPE, MAE, MAPE, MSE and RMSE in Regression Analysis Evaluation*, PeerJ. Comput. Sci., 7, e623.