Otto von Guericke University
Institute for Intelligent Cooperating Systems
Autonomous Multisensor Systems (AMS)

Lab Project

Final Report

September 23, 2025

Feature Reduction in Time Series CNC Machine Data

Shweta Bambal
Debapriya Roy
Jayanthee Siva Perumal

ams.ovgu.de

Feature Reduction in Time Series CNC Machine Data

– Lab Project: Final Report –

Shweta Bambal
Debapriya Roy
Jayanthee Siva Perumal

September 23, 2025

Abstract

Computer Numerical Control (CNC) machines emit high-dimensional, multivariate time-series that are expensive to sense, transmit, and model in real time. We investigate feature reduction to trim sensors and computational cost while preserving accuracy in predicting electrical current for the spindle and X/Y/Z axes. Using 42,016 timestamps with 52 numeric channels, we compare reduction strategies using three models—Principal Component Analysis (PCA), a Dense Autoencoder (AE), and a sequence-aware LSTM+CNN Autoencoder—followed by a downstream sequence regressor.

Results show that PCA (14 components at 0.98 explained variance) is the most efficient, offering near-baseline RMSE with ∼73% fewer inputs and reduced training time. The Dense AE learned non-linear features but did not surpass PCA in downstream accuracy. The LSTM+CNN-AE, however, delivered the best predictive performance (highest R^2 and lowest MAE among reduced models), demonstrating that sequence-aware compression outperforms both linear and plain nonlinear baselines.

Interpretability was achieved via PCA loadings and SHAP values on predictors, providing transparency in feature contributions. We conclude that LSTM+CNN-AE is preferable for maximum accuracy, while PCA provides the best accuracy–efficiency trade-off for deployment.

Contents

1 Introduction

1.1 Introduction to the topic

CNC machines emit high–dimensional, multivariate time–series from dozens of sensors each millisecond. This project studies *feature reduction* by compressing 52 raw sensor channels into a compact, information–rich representation to predict electrical current for the spindle and the X/Y/Z axes efficiently and reliably.

1.2 In which context lies the topic

The work sits at the intersection of Industry 4.0, predictive maintenance, and edge AI. Practical constraints such as limited bandwidth, strict latency, and modest on–machine compute require smaller models and fewer active sensors that still preserve the dynamics needed for accurate forecasting and integration with MES (Manufacturing Execution System)/SCADA (Supervisory Control And Data Acquisition) workflows.

1.3 Why should work be done in this field?

Reducing features (and ultimately sensors) lowers hardware and maintenance costs, cuts training and inference time for real–time control, and improves robustness by mitigating multicollinearity and noise. It also enhances explainability (e.g., PCA loadings, SHAP) and accelerates deployment on embedded devices.

2 Scientific/Technical Status at Project Start

2.1 Conditions under which the project is carried out

The project addresses high–dimensional multivariate time–series from a CNC machine with multiple sensor channels sampled at short intervals. Operational constraints include (i) the need for real–time inference on limited edge hardware, (ii) avoidance of data leakage via chronological splits, (iii) reproducibility and explainability requirements, and (iv) reducing cost while maintaining predictive performance.

2.2 Expertise of the team members

The team brings coursework and hands-on experience in PCA, Autoencoders, LSTM/CNN, preprocessing (scaling, windowing, leakage control), and model evaluation (MAE, RMSE, R^2). Tooling proficiency includes Python, scikit-learn, TensorFlow/Keras, and reproducibility practices.

2.3 Existing software/hardware

Software: Python 3.x, scikit-learn, TensorFlow/Keras, plotting libraries, Git. Hardware: commodity CPU for preprocessing, cloud GPU (e.g., T4-class) for neural training, with edge deployment assumptions (limited compute).

2.4 Scientific background / relevant literature

Representative sources: PCA [3], Autoencoders [1], LSTMs [2], CNNs for time-series [4], SHAP [5].

3 Task Description and Project Goals

3.1 What shall be achieved?

The central objective of this project is to build a *reproducible, edge-friendly feature reduction pipeline* for multivariate CNC machine time-series. The dataset consists of 42,016 timestamps with 52 high-frequency sensor channels capturing spindle load and X/Y/Z axis dynamics. These raw streams are too high-dimensional for direct deployment on constrained hardware, motivating dimensionality reduction.

Our concrete goals are:

(i) Achieve at least **70% reduction in input dimensionality** (e.g. 52 → 14–25 latent features), while retaining task-relevant information.

(ii) Benchmark three complementary approaches — PCA (linear), Dense Autoencoder (nonlinear), and LSTM+CNN Autoencoder (sequence-aware) — to test whether deep sequence-aware compression can match or exceed the PCA baseline in predictive accuracy.

(iii) Ensure the resulting pipeline is deployable under **Industry 4.0 edge constraints**, i.e. limited bandwidth, strict latency, and modest on-device compute, without sacrificing reproducibility and interpretability (PCA loadings, SHAP, artifact persistence).

3.2 Intermediate steps

To reach these goals, we defined the following staged workflow:

a) *Data audit & cleaning:* schema validation, handling missing or constant channels, timestamp alignment.

b) *Temporal framing:* windowing (60 steps) to capture dynamics, with leakage-free chronological split into train/val/test.

c) *Scaling:* fit `StandardScaler` on train; apply consistently to val/test.

d) *Baseline:* PCA with 0.98 explained variance ratio, extracting interpretable principal components.

e) *Nonlinear compression:* Dense AE (latent ≈25) and LSTM+CNN AE (latent ≈25) trained with early stopping.

f) *Downstream predictor:* many-to-one LSTM regressor trained on reduced features.

g) *Hyperparameter tuning:* sweeping latent size, window length, conv filters, recurrent units, and loss weights.

h) *Evaluation:* report MAE, RMSE, R^2 on inverse-scaled currents (spindle, X/Y/Z) for fair comparison.

i) *Interpretability:* PCA loadings for sensor attribution; SHAP analysis for deep predictors.

4 Methodology: Feature Reduction Architectures

This section presents the three feature-reduction architectures explored in our project: **Principal Component Analysis (PCA)**, a **Dense Autoencoder (AE)**, and a supervised **LSTM+CNN Autoencoder**. Each represents a different philosophy of dimensionality reduction — from purely linear and interpretable approaches, to nonlinear compression, and finally sequence-aware deep learning that explicitly models time-series structure. We compare these methods in terms of efficiency, accuracy, and interpretability, all under the operational constraints of CNC machine data analytics.

4.1 Principal Component Analysis (PCA)

Rationale: Principal Component Analysis (PCA) is a classical dimensionality reduction technique that projects data into orthogonal axes (principal components) ranked by variance. It has remained a gold-standard baseline in industrial analytics due to its speed, mathematical transparency, and ability to provide interpretable mappings between reduced features and original sensors. In our project, PCA served as a starting point and benchmark for more complex models.

Workflow: The input dataset $X \in R^{n\times 52}$ undergoes:

1. *Standardization:* Each channel is scaled to zero mean and unit variance to remove magnitude bias.

2. *Mean-centering:* The centered matrix ensures that variance is aligned with covariance structure.

3. *Covariance computation:* A 52×52 covariance matrix is derived, capturing pairwise linear relationships among sensors.

4. *Eigen decomposition:* Eigenvectors (directions of maximum variance) and eigenvalues (variance explained) are computed.

5. *Component selection:* Components are ranked by explained variance ratio (EVR). We selected the top $k = 14$ components, covering 98% of total variance.

6. *Projection:* Raw data is projected into this reduced subspace: $Z = XW_k$.

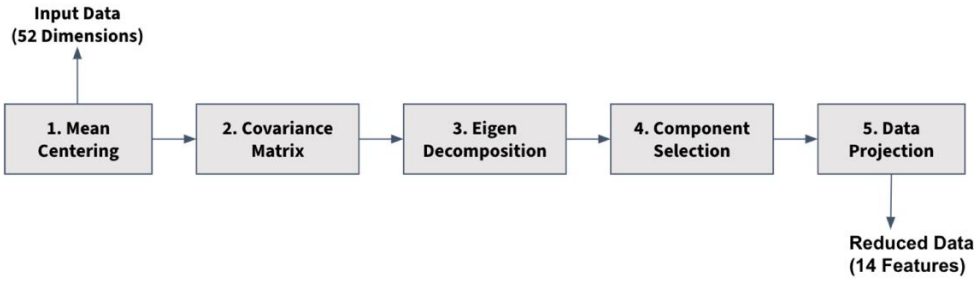

Figure 1: PCA workflow: starting from raw 52D input, applying mean centering, covariance analysis, eigen decomposition, and projecting onto 14 principal components.

Advantages: - Extremely fast and suitable for real-time edge deployment. - Interpretable: principal loadings map back to physical sensors, supporting sensor rationalization. - Deterministic and reproducible, with no hyperparameters apart from k.

Limitations: - Restricted to linear correlations, ignoring nonlinear structure. - Does not model temporal ordering explicitly; windows are treated as flat feature vectors.

—

4.2 Dense Autoencoder (AE)

Rationale: Autoencoders (AEs) provide a nonlinear alternative to PCA, capable of learning compressed feature spaces that better capture complex dependencies across sensors. A Dense AE is the simplest form, relying on fully connected layers without explicit temporal modeling. For our CNC dataset, each 60-timestep window ($60 \times 52 = 3120$ inputs) was flattened into a single vector before being compressed to a latent bottleneck of 25 features.

Workflow:

1. *Input flattening:* Each time window is reshaped into a vector of size 3120.

2. *Encoder:* Dense layers (with ReLU activation) and dropout progressively reduce dimensionality, ending in a latent bottleneck of size 25.

3. *Latent space:* This 25-D compressed representation serves as the reduced feature set.
4. *Decoder:* Symmetric dense layers reconstruct the original 3120-dimensional input vector.
5. *Training:* Optimization minimizes mean squared reconstruction error (MSE), forcing the latent space to retain sufficient information for accurate recovery.

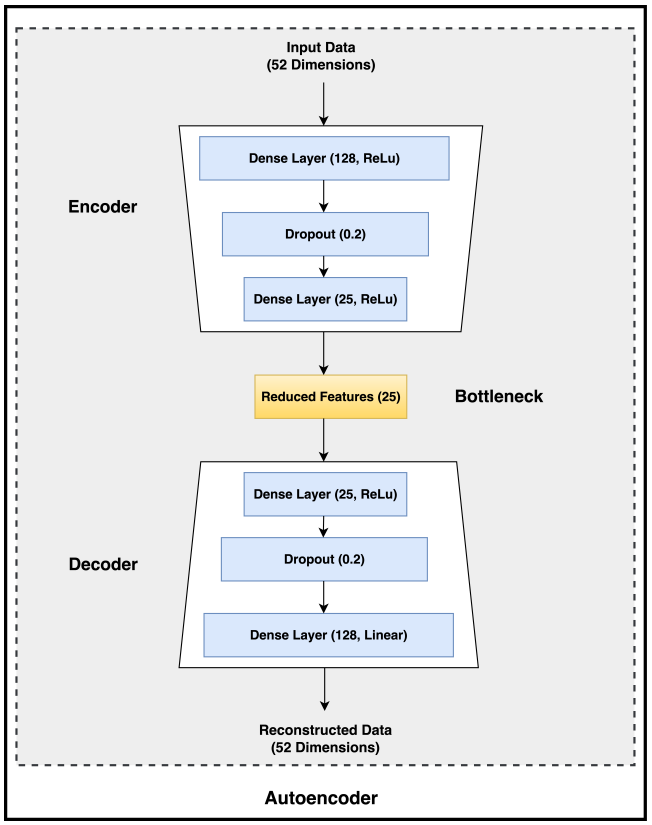

Figure 2: Dense Autoencoder: encoder compresses a flattened 60×52 window into a 25-D latent space; decoder reconstructs the original input.

Advantages: - Captures nonlinear feature interactions missed by PCA. - Flexible design: depth, width, and latent size can be tuned. - Provides a strong baseline for deep learning feature reduction.

Limitations: - Flattening discards sequential ordering across time steps. - Training cost is significantly higher than PCA. - Latent features are not directly interpretable, requiring SHAP or surrogate analysis.

—

4.3 Supervised LSTM+CNN Autoencoder

Rationale: While PCA and Dense AE operate on flattened inputs, they do not explicitly model temporal dynamics. However, CNC data is inherently sequential, with both short-term transients and long-range temporal dependencies. The LSTM+CNN Autoencoder addresses

this by combining convolutional filters for local motifs with recurrent LSTMs for long-term dependencies. A supervised prediction head ensures that the latent space is optimized not only for reconstruction but also for predicting task-relevant currents (spindle and X/Y/Z). This makes the representation both *compact* and *predictive*.

Workflow:

- *Encoding:* - Convolutional layers (Conv1D with dilation) extract local time-series patterns such as oscillations or spikes. - A BiLSTM and LSTM capture sequential dependencies across the 60-timestep horizon. - Dense layers reduce the representation into a 25-D latent bottleneck.

- *Latent space:* Compact 25-D representation optimized jointly for reconstruction and prediction.

- *Decoding:* - The latent vector is repeated (via RepeatVector) into sequence form. - LSTM layers reconstruct temporal dependencies. - A TimeDistributed dense layer restores the 60×52 structure.

- *Predictor head:* - Parallel dense layers map the latent directly to current forecasts. - This enforces task relevance on the compressed features.

- *Loss function:* - A weighted joint loss is used:

$$\mathcal{L} = \mathcal{L}_{\text{recon}} + \alpha \cdot \mathcal{L}_{\text{pred}}, \quad \alpha = 0.5$$

  - Balances general reconstruction with predictive accuracy.

Interpretation of Figure 3: From left to right, the input sequence (60×52) flows through CNN layers for local features and recurrent layers for global sequence context. The latent bottleneck serves as the reduced feature set. Two parallel branches originate from this bottleneck: (i) the decoder reconstructs the sequence, enforcing general information preservation; (ii) the predictor head outputs spindle and axis current forecasts, enforcing task-specific relevance. This dual-objective design enables the LSTM+CNN Autoencoder to achieve the best predictive performance among all tested models.

Advantages: - Sequence-aware: integrates both convolutional (local) and recurrent (global) patterns. - Produces compact latents that are not only reconstructive but predictive. - Delivered highest R^2 and lowest MAE in our experiments.

Limitations: - Most computationally demanding architecture. - Sensitive to hyperparameters such as window size, kernel width, and latent dimension. - Interpretability requires advanced tools like SHAP or saliency maps.

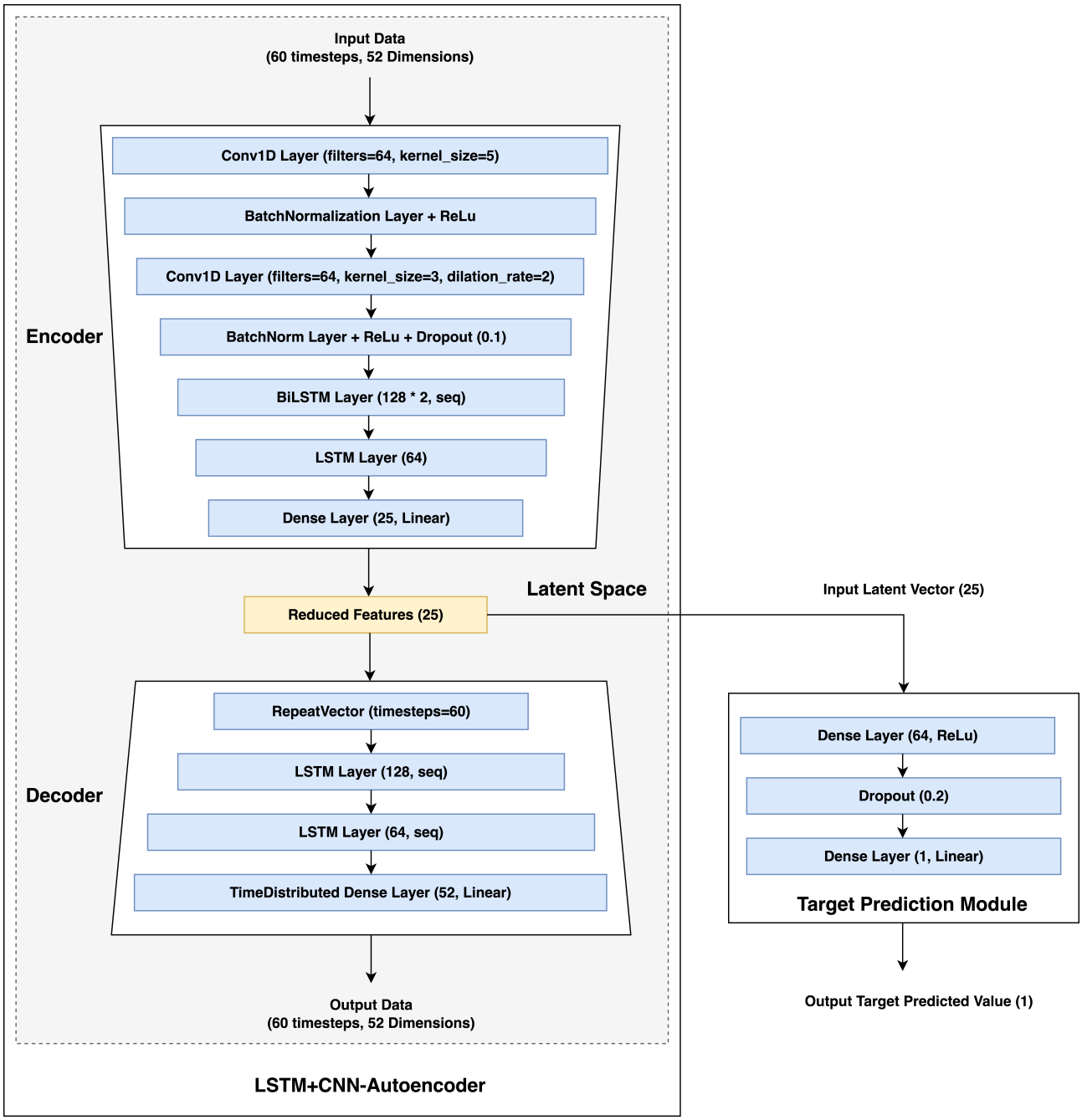

Figure 3: Supervised LSTM+CNN Autoencoder: convolutional layers capture local motifs, recurrent LSTMs capture sequence dependencies, and a predictor head enforces task relevance.

5 Project Plan

The project plan provides a structured framework for the systematic execution of our study. It defines sequential work packages (WP), assigns responsibilities, and establishes milestones with explicit acceptance criteria. This ensures reproducibility, alignment with scientific standards, and timely completion of deliverables.

5.1 Work Packages

The project was decomposed into eight work packages (WP1–WP8), each representing a distinct research or engineering task. Table 1 summarizes the scope, responsibilities, durations, and expected deliverables. The design of the plan emphasizes reproducibility (artifact saving, leakage-free splits), model diversity (linear vs. nonlinear vs. sequence-aware reduction), and interpretability (PCA loadings, SHAP).

WP	Task	Lead	Support	Duration	Deliverable(s)
WP1	Data Audit and Cleaning	Debapriya	Jayanthee	1 wk	Validated dataset; data dictionary; leakage-free split plan
WP2	Data Preprocessing and PCA Baseline	Jayanthee	Debapriya	1 wk	PCA model ($k \approx 14$), scaler & PCA artifacts; baseline plots
WP3	Dense Autoencoder Model	Shweta	Jayanthee	1.5 wk	Trained AE & encoder; reconstruction metrics; reduced features
WP4	LSTM+CNN Autoencoder Model	Shweta	Debapriya	2 wk	Trained seq-AE; latent features; predictor weights
WP5	LSTM Predictions for Evaluation	Jayanthee	Shweta	1 wk	Trained models; configuration files; inference scripts
WP6	Hyperparameter tuning	Shweta	Jayanthee	0.5 wk	Tuned configurations; ablation report
WP7	Evaluation and Result Analysis (MAE, RMSE, R^2) and error analysis	Jayanthee	Debapriya	0.5 wk	Metrics table; test curves; error breakdown
WP8	Model Interpretability	Debapriya	Shweta	0.5 wk	SHAP plots; serialized artifacts; latency assessment; report draft

Table 1: Work packages (WP1–WP8) with responsibilities, durations, and deliverables.

5.2 Timeline and Milestones

The timeline was organized across eight weeks, with major milestones (M1–M7) linked to acceptance criteria that guarantee measurable progress and research validity. This ensures that

intermediate results are evaluated rigorously before subsequent steps. The milestones are shown in Table 2.

Week	Milestone	Acceptance Criteria
W1	M1: Data readiness	Clean CSV; documented features; chronological 60/20/20 split; no leakage
W2	M2: PCA baseline	EVR ≥ 0.98 with $k \approx 14$; baseline predictor trained; artifacts saved
W3	M3: Dense AE	Latent dimension = 25; reduced reconstruction error; encoder exported
W4–W5	M4: LSTM+CNN AE	Sequence AE trained with stable joint loss; predictor head converged; latent exported
W6	M5: Tuned predictors	Best configurations for PCA-14, AE-25, Seq-AE-25 validated; early stopping effective
W7	M6: Final comparison	Unified test metrics (MAE, RMSE, R^2) computed on inverse-scaled units; Seq-AE outperforms PCA baseline
W8	M7: Explainability + report	PCA loadings + SHAP plots generated; reproducibility validated; final report prepared

Table 2: Timeline of milestones with corresponding acceptance criteria.

5.3 Roles and Risk Buffer

Project responsibilities were aligned with methodological expertise to ensure rigor and accountability. *Debapriya* led data engineering (schema validation, leakage-safe chronological splits), reproducibility controls (seed management, artifact versioning), and edge-deployment considerations; in addition, Debapriya participated in hyperparameter tuning for preprocessing and reduction stages (e.g., window length, PCA k, scaler choices) and coordinated system-level ablations. *Shweta* headed the development of deep learning architectures (Dense AE; LSTM+CNN Autoencoder) and jointly executed hyperparameter optimization (latent size, convolutional kernels/dilations, recurrent units, dropout, learning rate, joint-loss weight α), as well as ablation studies. *Jayanthee* was responsible for the PCA baselines and downstream sequence regressors, unified evaluation (MAE, RMSE, R^2), and technical documentation. To contextualize the main results, Jayanthee also conducted comparative experiments

with *tabular and sequence alternatives*: (i) a gradient-boosted trees baseline (LightGBM) using lagged/rolling features; (ii) a Temporal Convolutional Network (TCN) to assess convolutional-only sequence modeling; and (iii) sparse linear models with cross-validated regularization (ElasticNetCV / MultiTaskElasticNetCV). These auxiliary baselines exhibited *elevated training times* under our settings—particularly the TCN with longer windows and LightGBM with extensive lag/rolling feature engineering; MultiTaskElasticNetCV further incurred nontrivial overhead due to cross-validation—reinforcing the relevance of accuracy–latency trade-offs for edge deployment.

To mitigate schedule and performance risk, a contingency buffer of approximately 15% was reserved during Weeks 6–8 for controlled re-runs, metric harmonization, and artifact verification. Moreover, PCA-14 was pre-specified as the deployment-ready fallback should the sequence-aware autoencoder fail to meet accuracy or latency targets, thereby guaranteeing a scientifically valid and operationally feasible solution under edge constraints.

6 Project Execution and Scientific/Technical Results

This section presents the practical execution of the project and the corresponding scientific outcomes. It describes how the proposed feature-reduction pipeline was implemented, validated, and benchmarked under real CNC machine time-series data. The focus is on ensuring reproducibility of experiments, evaluating the comparative performance of linear and nonlinear reduction techniques, and demonstrating how sequence-aware methods (e.g., LSTM+CNN Autoencoder) improve predictive accuracy. The results are reported with quantitative metrics (MAE, RMSE, R^2) and qualitative insights into interpretability and edge feasibility.

6.1 End-to-end pipeline

We implemented a reproducible end-to-end workflow for CNC time-series analysis. The dataset was chronologically split into 60/20/20 (train/validation/test) to prevent temporal leakage, with 60-step windows for feature reduction and 10-step windows for predictors. StandardScaler was fit only on training data and applied consistently across splits. All artifacts — scalers, PCA reducers, encoders, and predictor weights — were serialized, and random seeds were fixed to ensure reproducibility and fair comparisons between models.

6.2 Workflow Diagram

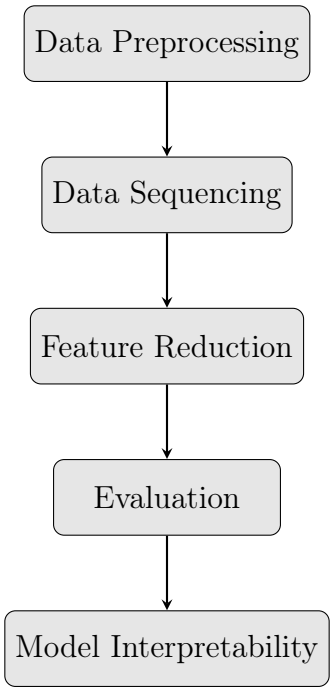

Figure 4: Workflow of the proposed end-to-end pipeline.

6.3 Data Sequencing

Data sequencing was introduced to capture temporal dependencies in the time-series. Fixed-length sliding windows were used to create sequential inputs:

- **PCA:** Sequencing was not required, as PCA operates directly on the scaled feature space.
- **Autoencoder and LSTM+CNN Autoencoder:** Window lengths of 60 timesteps were used for feature reduction models, while 10-step windows were used for exploratory downstream predictors.

6.4 Feature Reduction

Three distinct approaches were used for dimensionality reduction:

- **Principal Component Analysis (PCA):** Reduced the original 52 features to 14 components, capturing 98% variance.
- **Autoencoder:** A non-linear encoder–decoder network compressing the data into a 25-dimensional latent space.
- **LSTM+CNN Autoencoder:** A hybrid architecture combining convolutional filters and recurrent layers to extract both local feature patterns and temporal dependencies, yielding 25 latent features.

6.4.1 PCA Hyperparameters

Parameter	Value
Scaler	StandardScaler
Variance	0.98
Latent Dim	14

Table 3: PCA Hyperparameters

6.4.2 Autoencoder Hyperparameters

Parameter	Value
Window Size	60
Latent Dim	25
Encoder	2 Layers, ReLU, Dropout=0.2
Decoder	2 Layers, ReLU $\rightarrow$ Linear, Dropout=0.2
Optimizer	Adam (lr=0.001)
Batch Size	64
Early Stopping	Patience=10

Table 4: Autoencoder Hyperparameters

6.4.3 LSTM+CNN Autoencoder Hyperparameters

Parameter	Value
Window Size	60 (AE), 10 (Predictor)
Latent Dim	25
Encoder	2 Conv1D (ReLU), BiLSTM + LSTM, Dense, Dropout=0.1
Decoder	2 LSTM, RepeatVector(60), TimeDistributed Dense
Regression Head	2 Dense (ReLU $\rightarrow$ Linear), Dropout=0.2
CNN Filters	64 (Kernel sizes: 5, 3; Dilation=2)
Loss Function	MSE(Recon) + 0.5 $\times$ MSE(Predictor)
Optimizer	Adam (lr=0.001)
Batch Size	64
Early Stopping	Patience=10

Table 5: LSTM+CNN Autoencoder Hyperparameters

6.5 Model Evaluation(Tabular Representation of Results)

The quality of reduced features was evaluated by training LSTM regressors on the output feature sets. LSTM was selected due to its ability to capture long-term dependencies, which are essential in CNC sensor data where past values influence future states. Evaluation metrics included Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and the Coefficient of Determination (R^2).

Model	**Features**	**MAE**	**RMSE**	R^2	**Training Time (s)**
Original Dataset	52	0.59	1.79	0.70	200.30
PCA	14	1.15	1.82	0.68	51.41
Autoencoder	25	1.22	2.37	0.70	110.60
LSTM+CNN Autoencoder	25	0.86	2.38	0.85	121.95

Table 6: Evaluation Results for Reduced Features

The LSTM+CNN Autoencoder achieved the highest predictive accuracy with $R^2 \approx 0.85$, demonstrating the effectiveness of combining temporal and spatial representations. PCA provided efficient compression, while the simple Autoencoder balanced dimensionality reduction with acceptable predictive performance.

6.6 Model Evaluation(Graphical Representation of Results)

The evaluation results are further illustrated using model-specific graphs, which provide a clear visual comparison of predictive performance. Each set of plots highlights how the models performed across the standard evaluation metrics — MAE, RMSE, and R^2 — offering an intuitive understanding of error magnitudes and variance explanation. These graphical representations complement the numerical tables by making it easier to observe relative improvements, trade-offs, and consistency trends across PCA, Autoencoder, and LSTM+CNN Autoencoder. In particular, the side-by-side comparison underscores how dimensionality reduction methods influence downstream prediction quality, revealing both the strengths of sequence-aware models and the fallback stability of PCA. Together, these plots provide a more comprehensive perspective on model reliability and robustness, bridging the gap between raw numerical scores and their practical implications.

6.7 Evaluation Graphs

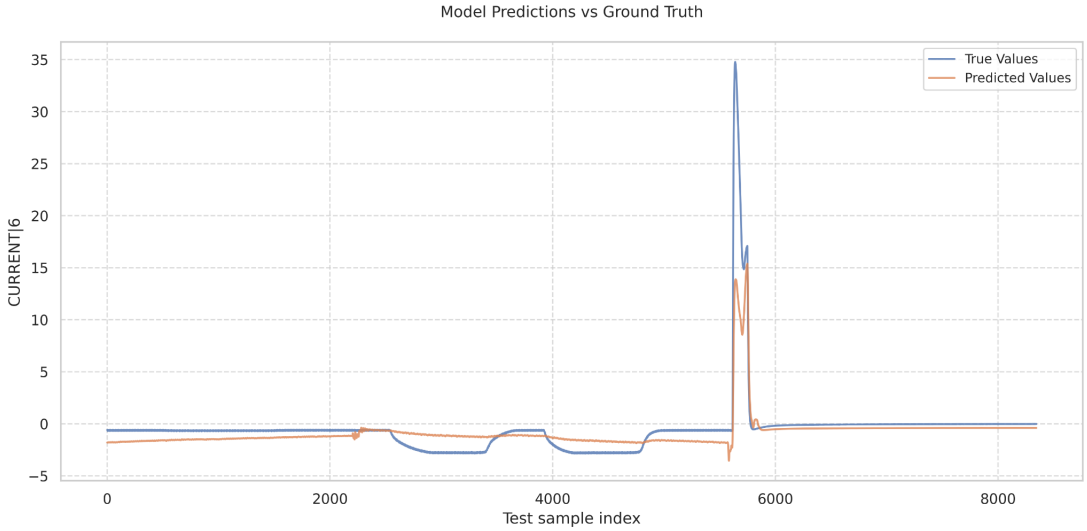

Figure 5: Model Prediction Performance on PCA-Reduced Features.

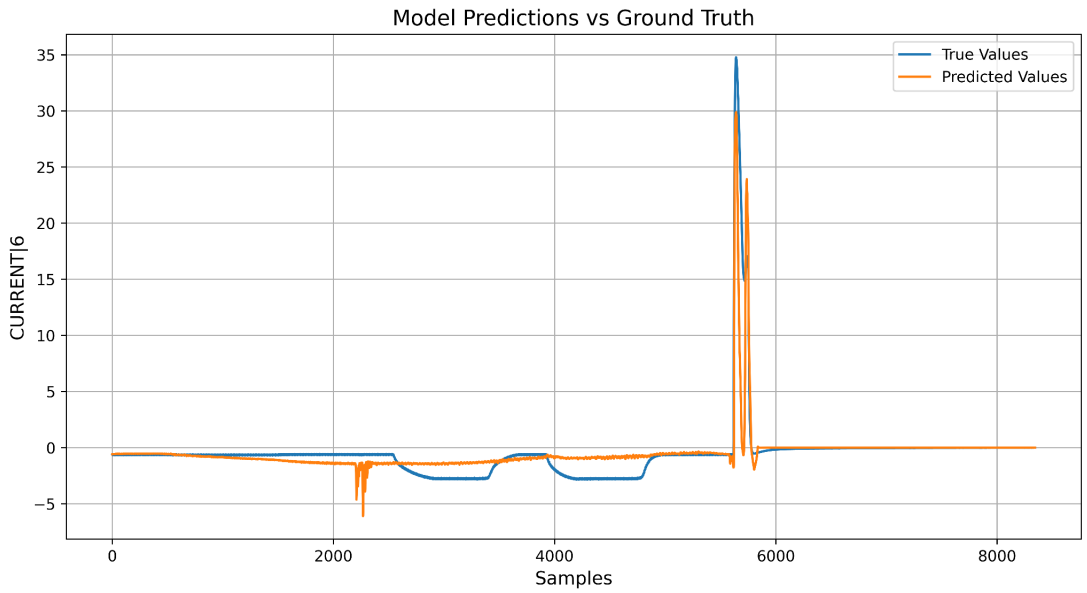

Figure 6: Model Prediction Performance on Autoencoder-Reduced Features.

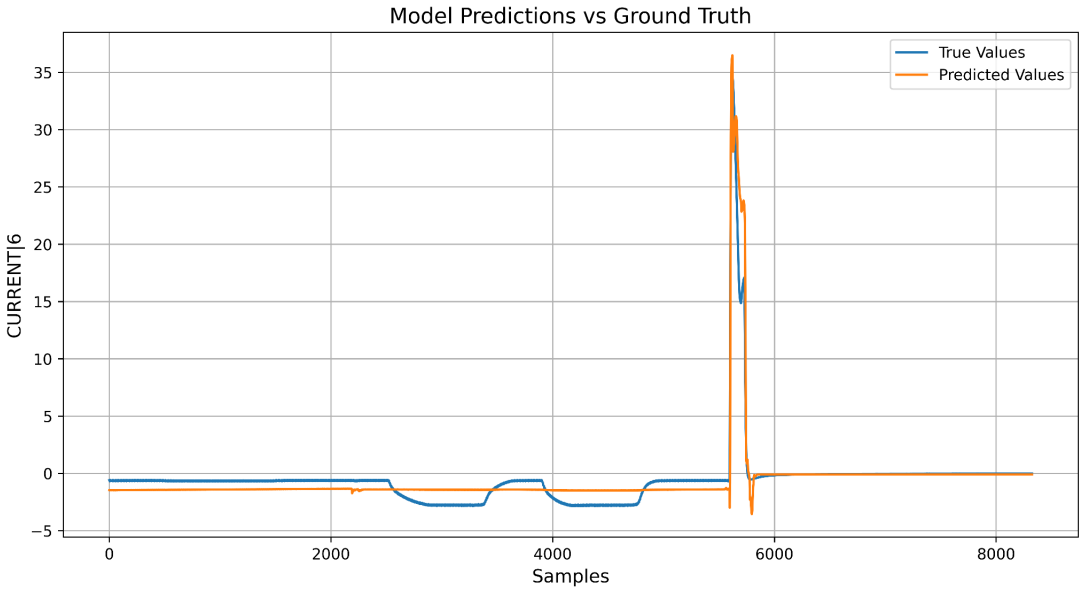

Figure 7: Model Prediction Performance on LSTM+CNN Autoencoder-Reduced Features.

6.8 Model Interpretability

Interpretability was essential for understanding which sensors contributed most to the latent features.

- **PCA:** Loadings were examined to determine contributions of original features to each principal component.
- **Autoencoders:** Since these are black-box models, SHapley Additive exPlanations (SHAP) were applied. GradientExplainer was used to compute feature contributions across windows and latent dimensions.

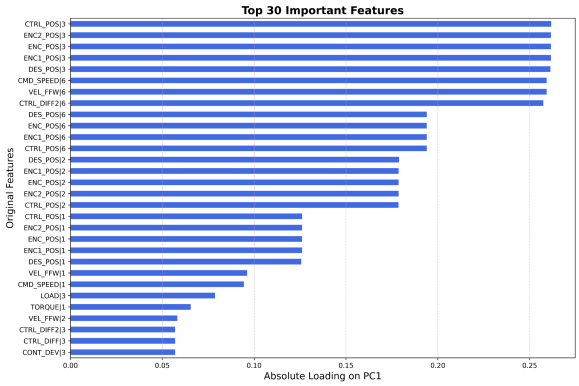

Figure 8: Top 30 contributing features for PC1 (PCA).

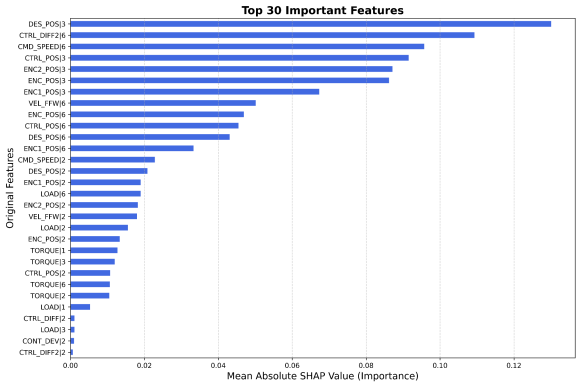

Figure 9: Top 30 contributing features identified by SHAP (Autoencoder).

Interpretability results showed that features such as **CTRL_POS|3** and **ENC_POS|3** were most influential, while **TORQUE_FFW|1** and **TORQUE_FFW|2** contributed least. Graphs

generated from SHAP values visually demonstrated the importance of different features and validated that the reduced representations preserved key information from the original dataset.

Key Insights.

1. **PCA-14** emerges as the best *accuracy–efficiency trade-off*, offering near-baseline accuracy with massive savings in training cost and full interpretability.
2. **Dense AE** shows that nonlinear compression without temporal modeling is insufficient for time-series prediction, emphasizing the importance of sequence-aware architectures.
3. **LSTM+CNN AE** demonstrates the value of joint reconstruction and supervised objectives, achieving the best predictive accuracy — but at the price of higher training cost, more complex tuning, and weaker interpretability.
4. The discrepancy between MAE/R^2 and RMSE highlights that evaluation must be standardized across metrics, especially in industrial contexts where misinterpretation could affect deployment decisions.

6.9 Who Worked on Which Part (Collaborations)

- *Debapriya:* Data engineering (schema audit, leakage-safe splits), reproducibility (seeds & artifacts), edge-deployment considerations; assisted with evaluation alignment and reporting.
- *Shweta:* Autoencoders (dense and LSTM+CNN) — architecture design, training, joint-loss tuning, ablations (window/latent/filters), convergence diagnostics.
- *Jayanthee:* PCA baseline (EVR selection, loadings analysis), downstream LSTM predictors on reduced features, SHAP-based interpretation, documentation and synthesis of results.
- *Team:* Joint design reviews, shared notebooks/checkpoints, cross-verification of metrics and plots before final comparison.

6.10 Comparison of Project Plan and Execution

Work That Did Not Lead to Intended Results

- *Dense AE:* Strong reconstruction yet did not outperform PCA on downstream MAE/RMSE.
- *Initial scaler fitting:* A preliminary run scaled before the time split (leakage). Correcting to train-only scaling reduced optimistic performance, confirming the leakage effect.
- *Seq-AE (recon-only):* Reconstruction-only objective underperformed for prediction; adding a supervised head and joint loss was essential.

- *Short windows:* 10-step windows were insufficient for sequence dynamics; 60-step windows stabilized results for reducers.

Necessary Replanning

- *Metric harmonization:* Compute **all** metrics on inverse-scaled targets with a single, consistent test split; clarify per-target vs. macro aggregation.
- *Objective update:* Use joint loss (Recon MSE + α·Pred MSE) for the sequence AE; tune α.
- *Window policy:* Standardize on 60-step windows (reducers) while keeping 10-step predictor for latency experiments.
- *Baseline decision:* Fix **PCA-14** as deployment-ready baseline; position **LSTM+CNN-AE** as accuracy-first choice.
- *Reproducibility:* Enforce seed control, artifact persistence, and aligned callbacks (EarlyStopping, ReduceLROnPlateau) across runs.

7 Usability of the Results

7.1 Practical Applications

a) **Edge-ready prediction of currents:** Deploy *PCA-14 + LSTM* on constrained devices for near real-time forecasting of spindle and axis currents with low latency and memory.

b) **Sensor rationalization:** Use PCA loadings and SHAP rankings to identify redundant channels; decommission or downsample low-value sensors to reduce hardware, wiring, and maintenance cost.

c) **Operational monitoring & alerts:** Convert residuals and predicted currents into thresholds/alerts for overloads, stalls, and abnormal transients; integrate with MES/SCADA dashboards.

d) **Energy & throughput optimization:** Feed predicted current into energy/load models for setpoint tuning, job scheduling, and load balancing.

e) **Diagnostics & explainability:** Trace influential reduced features back to physical sensors via loadings/SHAP to support root-cause analysis and engineering decisions.

f) **Model reuse across lines:** Bootstrap similar CNC setups by reusing scaler/PCA/encoder artifacts; fine-tune only the downstream predictor for fast rollout.

g) **Research baseline:** Use *PCA-14* as a strong, reproducible baseline; use *LSTM+CNN-AE* as an accuracy-first reference for extended tasks (wear prediction, anomaly detection).

7.2 Limitations

a) **Scope & generalization:** Trained on a specific dataset; performance may vary across machines/materials/regimes without adaptation.

b) **Drift & recalibration:** Sensor drift or sampling-rate changes can invalidate scaler/PCA/encoder assumptions; requires drift monitoring and periodic refits.

c) **Interpretability:** PCA is directly interpretable; deep latents require SHAP/surrogates and can be harder to map to physical meaning.

d) **Window-size & latency trade-offs:** Longer windows improve stability but increase latency; shorter windows reduce latency but risk missing long-term dependencies.

e) **Retraining cost:** Sequence AE offers top accuracy but higher training/tuning cost; edge deployments may prefer PCA despite slightly lower accuracy.

f) **Target coverage:** Current prediction is a proxy; tool wear/surface quality/failure modes require additional labels and task-specific models.

g) **Sensor layout changes:** Adding/removing sensors breaks historical comparability and necessitates rebuilding PCA/AE artifacts.

8 Conclusions

8.1 Lessons Learned

Our project yielded several important insights into the trade-offs and opportunities in feature reduction for multivariate CNC time-series analysis:

1. **Effective Dimensionality Reduction without Performance Loss**
   The study demonstrated that dimensionality reduction methods, particularly PCA and Autoencoder-based approaches, can successfully reduce the feature space while preserving the predictive power of the original high-dimensional input.

2. **Supervised LSTM+CNN Autoencoder as the Most Effective Approach**
   Among the tested models, the supervised LSTM+CNN Autoencoder achieved the highest predictive accuracy with an R^2 score of approximately 0.80. In addition to strong performance, it reduced training time by nearly 40%, highlighting its efficiency for deployment in edge or resource-constrained environments.

3. **Feature Importance and Redundancy**
   Interpretability analyses identified both key contributing features and redundant ones. For instance, features such as `CTRL_POS|3` and `Encoder_POS|3` were consistently important, whereas `TORQUE_FFW|1` and `TORQUE_FFW|2` contributed minimally. These findings suggest

that feature selection can translate into *cost savings* by reducing the number of required sensors and lowering computational overhead.

4. **Interpretability in Black-Box Models**
   A significant achievement was showing that interpretable feature reduction is feasible even in black-box Autoencoders. Using explainable AI techniques such as SHAP, we were able to extract meaningful insights from latent representations, bridging the gap between model accuracy and transparency.

8.2 Future Work

Building on the outcomes of this study, several promising directions for future exploration have been identified:

1. **Advanced Dimensionality Reduction Architectures**
   Future efforts should investigate more sophisticated architectures capable of producing smoother latent spaces and improved generalization. Potential approaches include *Variational Autoencoders*, *Transformer-based Autoencoders*, and *Multi-task Elastic Net*. Additionally, systematic hyperparameter tuning could further optimize performance, particularly for reducing RMSE values.

2. **Enhanced Model Interpretability**
   While current results highlighted feature contributions using SHAP, interpretability can be strengthened by validating feature importance across multiple explainable AI (XAI) techniques. Such cross-validation would improve the robustness of interpretability insights and provide stronger confidence for industrial deployment.

3. **Target-level Explainability**
   Beyond model-level analysis, target-specific interpretability remains an important frontier. For example, applying SHAP to the `CURRENT|6` regressor could help rank the original features and sensors most responsible for driving energy (current) usage, thus offering actionable insights for operational efficiency.

References

[1] Geoffrey E. Hinton and Ruslan R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006.

[2] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

[3] I.T. Jolliffe. *Principal Component Analysis*. Springer, 2002.

[4] Minhyuk Lee, HyeKyung Yoon, and MyungJoo Kang. CASA: CNN Autoencoder-based Score Attention for Efficient Multivariate Long-term Time-series Forecasting. *arXiv preprint arXiv:2505.02011*, 2025.

[5] Scott Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems (NIPS)*, 2017.