(Online) ISSN 2712-0562
Original Research

Sustainable Engineering and Innovation
Vol. 8, No. 2, 2026, pp.145-160
https://doi.org/10.37868/sei.v8i2.id1019

Application of parallel computing in forecasting problems of complex engineering processes

Vladyslav Kozub[1*], Yurii Shevchuk[2], Orest Danchak[3], Yevhenii Tytarchuk[4], Anatolii Subin[5]

[1] Department of Information Technology and Systems, Luhansk Taras Shevchenko National University, Lubny, Ukraine
[2] Momentum3, Tulsa, United States
[3] Department of Artificial Intelligence Systems, Lviv Polytechnic National University, Lviv, Ukraine
[4] Department of Computer Sciences and Digital Economics, Vinnytsia National Agrarian University, Vinnytsia, Ukraine
[5] Department of Mechanical Engineering Technology, Institute of Mechanical Engineering, National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute", Kyiv, Ukraine

*Corresponding author E-mail: kozubvuifmit@luguniv.edu.ua

Received Jun. 7, 2026
Revised Aug. 29, 2026
Accepted Sep. 14, 2026
Online Sep. 29, 2026

Abstract

Complex engineering processes require short-term prediction to maintain the stability of energy-intensive industrial operations under severe computational and latency constraints. The research is a 24-hour-ahead multi-step prediction of aggregated electrical load in one of the Ukrainian metallurgical plants, where high-frequency SCADA data are sampled at 15-minute intervals, totaling 105,120 observations over 2022-2025. This is to predict energy demand with a maximum inference latency of 250 ms to enable real-time dispatch coordination. It is an advanced model that uses mixed-precision (FP16) computing and asynchronous data loading on GPU-accelerated data parallelism (CUDA 12.2, PyTorch 2.1, NVIDIA RTX A6000 48 GB). A leakage-conscious walk-forward validation protocol performs five repeated validations and guarantees temporal consistency and reproducibility. Forecast performance is evaluated using mean absolute error (MAE), root mean squared error (RMSE), mean absolute percentage error (MAPE), and R^2. At the same time, computational efficiency is assessed in terms of speedup, throughput, and inference latency. The proposed parallel hybrid framework reduces the MAE to 2.41 MW, the RMSE to 3.05 MW, the MAPE to 2.97%, and the R^2 to 0.98, outperforming the ARIMA, standalone LSTM, and Transformer baselines. GPU acceleration achieves a 6.67-fold reduction in training time, increases throughput from 415 to 1964 samples per second, and lowers inference latency from 312 ms to 87 ms, satisfying operational constraints. Parallel deep forecasting can support real-time industrial decisions, improve load balancing, and enhance energy resilience in Ukraine.

Keywords: Parallel computing, Engineering forecasting, GPU acceleration, Time-Series deep learning, Industrial energy systems

1. Introduction

Engineering processes deployed in heavy industrial environments are governed by complex nonlinear temporal dynamics arising from interactions among thermodynamic behavior, mechanical loading conditions, environmental disturbances, and automated operational control mechanisms. Reliable forecasting of such processes has therefore become essential for predictive maintenance scheduling, production planning, adaptive load management, and real-time operational risk mitigation in energy-intensive industrial systems [1].

The increasing adoption of Supervisory Control and Data Acquisition (SCADA) systems and industrial cyber-physical infrastructures has significantly increased the volume and velocity of operational data streams, creating a growing need for computational models capable of capturing multi-scale temporal dependencies embedded in high-frequency engineering process data [2].

In the industrial case, traditional forecasting methods have, to a large extent, relied on statistical time-series models, with their mathematical interpretability and established theoretical basis. Groups such as autoregressive integrated moving average (ARIMA) and exponential smoothing methods are classical methods that have been extensively used in engineering for prediction tasks, under the assumption of linearity and weak stationarity [3].

Nonlinear dependence, regime variability, and exogenous disturbances, however, often occur in real-world industrial process data and are inconsistent with these assumptions, making the predictive robustness of traditional statistical models in complex production settings weak [4], [5], [6].

Experimental studies also indicate that the classical linear model is inadequate in most instances for modeling hierarchical temporal dependencies in multivariate industrial sensor data, and hence leads to degraded forecasting capability in a dynamic operational environment [7], [8].

The recent developments in machine learning and deep learning have also brought about a radical change in the forecasting capabilities of nonlinear time-series modeling. In engineering systems, recurrent neural networks (RNNs) have been successful in capturing long-range temporal relationships and sequential dependencies, particularly long short-term memory (LSTM) networks [9].

Similarly, attention-based sequence modeling models like Transformers allow for modeling global contextual interactions using self-attention and have been shown to work better for long-horizon predictions [10-12]. In fact, recurrent memory architectures combined with attention mechanisms have proven useful for addressing industrial forecasting tasks like energy demand forecasting, process monitoring, and load forecasting, as mentioned in [13] and [14].

Deep sequential architectures provide predictive advantages, but also cause significant compute cost during training and inference. The optimization of high-dimensional neural networks is complicated, and they demand significant numerical computation and memory; thus, they are not often exploited for real-time industrial decision-support systems, where they have to comply with a latency requirement [15-17].

These limitations have led people to think of parallel computing and high-performance computing (HPC) architectures to speed up machine learning workloads and make them more scalable [18]. The use of a computing environment with GPUs enables the parallel execution of large numbers of numerical operations, thereby increasing computational throughput and minimizing training time for models used in data-intensive forecasting problems [19-21].

In addition, distributed computing infrastructures can be exploited to provide increased scalability, where workload can be distributed and activities optimized across a variety of different processing units [22-25]. The use of deep learning designs for parallel computing designs has been demonstrated to have great potential to improve the prediction performance and computational efficiency in engineering forecasting applications [26].

The parallelized forecasting architectures are effective for modelling rich multivariate dependencies that are suitable for the noisy, non-stationary industrial environment [27]. In energy-intensive industries, like

metallurgical production or industrial manufacturing, the accuracy of the forecasting of the process variables and energy consumption is crucial to improve operational stability, resource usage, and production efficiency [28], [29].

However, the data collected from industrial processes using SCADA systems typically contain missing data, imprecise measurements, and different time scales; therefore, there is a need for proper pre-processing and validation operations prior to the model's implementation [30-33]. Furthermore, real-time industries demand tight operational constraints like low-latency inference [34] and scalability of forecasting pipelines in an automated control system [35].

The present literature has shown that hybrid deep learning architectures and parallel computing systems are effective in enhancing forecasting accuracy, but there are important challenges that need to be overcome.

The majority of the existing techniques focus more on prediction accuracy than on computer scalability in real-time operating conditions. In contrast, parallel computing research is more likely to be concerned with computing efficiency and is less likely to provide a systematic evaluation of the accuracy of forecasts in an industrial process context. Furthermore, few studies have investigated the verifiability of reproducible, GPU-accelerated hybrid forecasting models, which are tested with real industrial production data under strict test conditions with limited latency and leakage [36-39].

From the above discussion, it is worth noting that generally there is a serious problem with the design of an integrated forecasting framework that can guarantee high predictive accuracy, computational scalability, and real-time deployment of complex engineering processes under dynamic industrial conditions.

The existing methods either strive for the best prediction precision with no emphasis on the thresholds of operational latency or speed up computations without considering the reliability of the predictions in the actual production environment.

In this paper, a scalable and efficient forecasting architecture is introduced that addresses the real-time performance requirements of a firm while preserving the nonlinear behavior of engineering processes. This paper then presents the development of a scalable, computationally efficient forecasting structure that is capable of capturing the nonlinear behavior of engineering processes while satisfying a firm's real-time performance requirements in an industrial context [40].

To solve this issue, the current paper provides the following contributions:

i. A parallel deep learning model with long short-term memory (LSTM) networks and Transformer-based attention mechanisms to model complex nonlinear engineering time series.

ii. Application of GPU-based data parallelism and mixed-precision training to realize high computational efficiency and low-latency inference for industrial deployment.

iii. A time-series assessment model that used chronological subdivision and walk-forward validation conformed to the operational characteristics of industrial SCADA.

iv. Full experimental validation based on real data of the metallurgical process, showing an improvement in accuracy and computational performance.

By combining modern deep learning architectures with parallel computing solutions, this research can enhance the predictive accuracy and computational scalability of complex engineering forecasting tasks and enable real-time operation in industrial decision-support environments.

2. Research method

The data used in this study, in terms of operations, were obtained from the Supervisory Control and Data Acquisition (SCADA) infrastructure of an industrial metallurgical production site located in Central Ukraine.

The SCADA system is a multivariate system that continuously records process variables such as electrical load demand, environmental conditions, thermodynamic conditions, production scheduling indicators, and equipment operational parameters. These variables are high-dimensional, time-dependent, complex data, meaning that industry dynamics are nonlinear, production consists of cyclical regimes, and operational disturbances are stochastic, all of which are characteristic of real-world engineering conditions.

A high-quality data analysis showed that the dataset had 3.84% missing data, which were not evenly distributed across time sequences, due to latency in communication, sensor malfunctions, and hardware-related interruptions. A two-stage imputation strategy was adopted to maintain temporal continuity and avoid information leakage.

Gaps that were less than four consecutive intervals were filled in through linear interpolation, and other longer gaps were filled in by imputation with seasonal moving averages of pure historical data. The reconstruction procedures were performed solely on past information because they had to be methodologically reproducible and minimize leakage.

A data dictionary (Table 1) was developed to further clarify the data and to ensure the data are reproducible. The observed ranges of the variables in the study period are electrical load demand (12.4-87.6 MW); ambient temperature (18.2-36.7 degC); furnace pressure (0.91-2.84 MPa); production index (0.00-1.00); and energy consumption rate (5.1-42.3 MWh). Certain SCADA information is not publicly available because of confidentiality agreements, security policies, etc.

Table 1. Data dictionary

Variable Name	Description	Data Type	Unit	Role
Time Index	Timestamp or observation index	Integer	Time step	Identifier
Temperature	System operating temperature	Float	°C	Input
Pressure	Process pressure level	Float	Pa	Input
Flow Rate	Material or energy flow rate	Float	m^3/s	Input
Load	System operational load	Float	%	Input
Vibration	Mechanical vibration level	Float	Hz	Input
Energy Consumption	Energy usage of the system	Float	kWh	Input
System Output	Observed system performance measure	Float	-	Target
Forecast Value	Predicted output value	Float	-	Output

All datasets were anonymized, as any information about the person was removed, as well as information about the sensors and any proprietary operational times. However, the data is only available to industry partners via partnership agreements, but preprocessing pipelines and model configurations can be reproduced upon request.

For all continuous variables, we applied z-score standardization, which was performed on the training partition only to guarantee numerical stability and the same scaling of the features. Normalization formulation is given by:

$$x' = \frac{(x - \mu_{train})}{\sigma_{train}}, \quad (1)$$

where μ train, and α train refer to the estimated mean and standard deviation using the training subset only.

The forecasting problem is defined as a direct multi-step time-series prediction problem to forecast aggregated electrical load over the 24-hour horizon, equivalent to 96 future intervals. Assume the multivariate input sequence at time t to be defined as:

$$X(t) = \{x^1(t), x^2(t), \ldots, x^n(t)\}. \quad (2)$$

The forecasting functionality is stated as:

$$\hat{y}_{\{t+1:t+h\}} = f_{\theta(X_{\{t-k+1:t\}})}. \quad (3)$$

In which $k = 96$ is the historical input window, $h = 96$ is the multi-step forecasting horizon, and $f(*)$ is the nonlinear forecasting model. A direct multi-output forecasting strategy is implemented rather than a recursive one to avoid cumulative error propagation across sequential steps.

The model supports both batch training and online inference to meet real-time industrial deployment requirements. The optimization problem is formulated with the help of Mean Squared Error (MSE) loss:

$$L(\theta) = (1/N)\,\Sigma\{i = 1\}^{\{N\}} || y_i - \hat{y}_i ||^2, \quad (4)$$

where N is the number of training samples.

Feature engineering was performed to improve the predictive performance of nonlinear dynamics in SCADA signals. To ensure complete 24-hour historical dependencies, temporal lag variables as long as 96-time steps were created. Rolling statistical measures, such as moving averages and rolling standard deviations, have been computed across several time windows to indicate short- and medium-term operational variations. The rate-of-change effect of process dynamics was modeled using first-order differencing, and periodic production regimes were modeled using cyclical encodings of temporal indices via sine-cosine transformations. The transformations that these models allow deep learning architectures to learn hierarchical temporal dependencies and regime changes found in industrial process data.

More advanced multi-GPU architectures, such as PyTorch Fully Shared Data Parallel (FSDP), exhibit almost linear scaling [41]. Comparative studies of models, including Horovod and DeepSpeed, reveal marked differences in throughput for large-scale distributed learning [42].

Newer GPU scheduling techniques enhance memory access and throughput for deep learning workloads [43]. PerfTop is a communication-computation overlap model that forecasts performance in distributed GPU training topologies [44].

Given the computational requirements of deep sequential architectures, the forecasting framework was implemented on a high-performance computing platform with GPUs.

The system architecture was a CPU with 12 cores and 2.4 GHz (Intel Xeon Silver 4214R), 128 GB of DDR4 RAM, and an NVIDIA RTX A6000 with 48 GB VRAM, running CUDA 12.2 and cuDNN 8.9. The backend system has been set up using Docker Compose [45] with PyTorch, CUDA, and NCCL for parallel communication, and a batch size of 64 has been chosen.

FP16 training was used to minimize memory usage and speed up numerical calculations, without degrading predictive performance.

Informally, the world data is divided between k computing nodes as:

$$F_{hybrid}(Xt) = F_{LSTM}(Xt) \cup F_{Transformer}(Xt). \quad (5)$$

Each partition undergoes a nonlinear transformation through the hybrid deep architecture:

$$Z\_i = \varphi(X\_i). \quad (6)$$

Local forecasts are generated as:

$$\hat{y}_{i(t+h)} = f_{i(Z_{i(t)})}. \quad (7)$$

The aggregated global prediction is defined as:

$$\hat{y}(t + h) = \Sigma_{\{i=1\}_i^{\{k\}} w_{\hat{y}_{i(t+h)}}}. \quad (8)$$

Parallel scalability is evaluated using standard speedup and efficiency metrics:

$$S = \frac{T_{CPU}}{T_{GPU}}; \quad (9)$$

$$E = \frac{S}{P}. \quad (10)$$

where *S* is the speedup, *E* is the parallel efficiency, and *P* is the number of computing units.

A single-GPU solution has been selected to provide a compromise between scalability, cost, and typical industrial control deployments where multi-GPU solutions have proven impractical because of the synchronization overhead and deployability issues.

Benchmarking was done at two levels with rigorous comparative evaluation. A classical autoregressive integrated moving average (ARIMA) model has been used as the statistical baseline, and more sophisticated deep learning models, such as standalone long short-term memory (LSTM), Transformer encoder, and the hybrid LSTM-transformer architecture, have been proposed. The hybrid model combines stacked LSTM layers to model short-term temporal dependencies, and a self-attention encoder to capture long-range interactions among contextual variables across multivariate sequences.

The Adam optimization algorithm with an initial learning rate of 1 × 10^-3, early stopping after 10 epochs, and gradient clipping with a norm threshold of 1.0 was used to optimize the model parameters and achieve stable convergence. To preserve temporal causality and avoid data leakage, the data were chronologically split into 70% training, 15% validation, and 15% testing sets without random shuffling.

The walk-forward validation plan was used, comprising five consecutive folds to account for realistic industrial forecasting scenarios when using growing training windows. Random seed fixation was performed to ensure reproducibility across all experimental environments (Python 42) [46], NumPy (42), and PyTorch (42). The performance results are presented as mean ± standard deviation values from five independent experimental runs to provide statistical strength. Moreover, bootstrap resampling was used to estimate 95% confidence intervals to determine predictive reliability within operational variability.

2.1. Ablation study

Three-ordered ablation experiments were performed to measure the contribution of architectural design, computational accuracy, and feature engineering. First, precision ablation was used to compare FP32 and FP16 training modes and assess the trade-off between computational acceleration and numerical precision. To ensure convergence stability and predictive metrics, predictive accuracy was evaluated to assess whether mixed precision affects it.

Second, structural ablation was performed on the standalone LSTM, the standalone transformer, and the proposed hybrid LSTM-transformer architecture, using the same preprocessing and hyperparameter settings. This study removes the contribution of attention-enhanced recurrent integration.

Third, feature ablation was used when comparing models trained on unprocessed SCADA inputs with models trained on engineered time representations. In this experiment, the incremental predictive utility of domain-guided feature transformations is tested.

These analyses of the ablation ensure that the subsequent sections can attribute their performance gains to systematic methodological improvements rather than incidental implementation effects. To assess performance, the standard quantitative measures of predictive accuracy in engineering time-series forecasting were mean absolute error (MAE), root mean squared error (RMSE), and mean absolute percentage error (MAPE).

In addition to statistical accuracy, computational scalability was measured by training time, throughput, inference latency, and parallel speedup. The optimized model was implemented on the GPU-enabled system to enable in-time predictions of the complex dynamics of industrial processes, thereby guaranteeing adherence to

predictive performance, predictive reproducibility, predictive scalability, predictive leakage, and strict predictive operational latency specifications of the intelligent industrial decision-support system [47].

2.1.1. Algorithm

The proposed algorithm is based on the GPU-accelerated hybrid LSTM–transformer forecasting framework for SCADA-based industrial load prediction. Let the input multivariate time-series dataset be denoted as *X(t)*, where each observation at time t is represented as a feature vector in a high-dimensional engineering state space.

$$X(t) = \{x_1(t), x_2(t), \dots, x_n(t)\}. \quad (11)$$

The preprocessed dataset is partitioned across parallel computational nodes, with each processing unit handling a subspace of the global dataset, as defined below.

$$X = \bigcup_{i=1}^{k} X_i. \quad (12)$$

Each subspace undergoes a localized feature transformation via a nonlinear mapping function to enhance its predictive representation capacity across engineering states.

$$Z_i = \varphi(X_i). \quad (13)$$

A parallel learning model is trained independently across distributed nodes, where the forecasting function for each subspace is defined as:

$$\hat{y}_i(t + h) = f_i(Z_i(t)). \quad (14)$$

The global forecasting model integrates all localized predictive outputs through an aggregation function that minimizes forecasting error across the distributed environment.

$$\hat{y}(t + h) = \sum_{i=1}^{k} w_i \hat{y}_i(t + h). \quad (15)$$

The optimization objective is to minimize the mean squared forecasting error across all computational nodes within the parallel system.

$$L = \left(\frac{1}{N}\right) \sum_{t=1}^{N} (y(t + h) - \hat{y}(t + h))^2. \quad (16)$$

The optimal forecasting parameters are obtained through iterative convergence using distributed gradient-based optimization techniques defined as:

$$\theta * = \arg\min L(\theta). \quad (17)$$

The final optimized model is deployed for real-time forecasting of complex engineering process dynamics within the parallel computing environment. The system architecture of the proposed algorithm is presented in Figure 1.

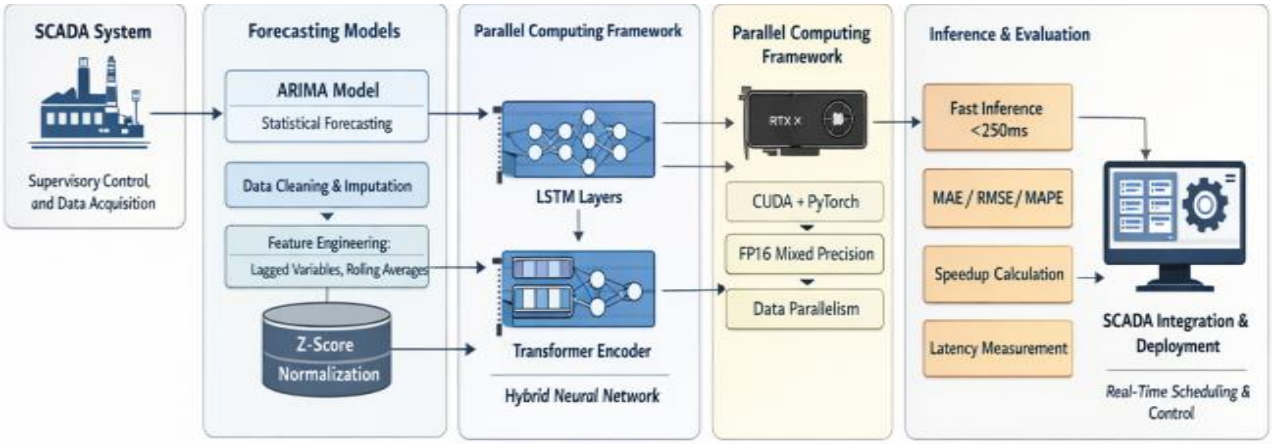

Figure 1. System architecture of the work

3. Results and discussion

This part presents an empirical test of the suggested hybrid LSTM-Transformer forecasting model implemented on a parallel computing system with a GPU. The metallurgical load dataset in the methodology was experimented on using a walk-forward validation protocol with a 24-hour forecast horizon of 96 future time steps. Two aspects are considered by the evaluation: (i) forecasting accuracy and (ii) computational performance. The predictive accuracy of the four competing models, which are ARIMA, standalone LSTM, standalone Transformer, and the proposed hybrid LSTM-Transformer model, is provided in Table 2.

Table 2. Forecasting accuracy comparison across competing models

Model	MAE (MW)	RMSE (MW)	MAPE (%)	R^2
ARIMA	4.82	6.14	5.76	0.91
LSTM	3.47	4.32	4.11	0.95
Transformer	3.12	3.98	3.84	0.96
Hybrid LSTM–Transformer	2.41	3.05	2.97	0.98

The performance is measured in terms of mean absolute error (MAE), root mean squared error (RMSE), mean absolute percentage error (MAPE), and coefficient of determination (R2) (Figure 2).

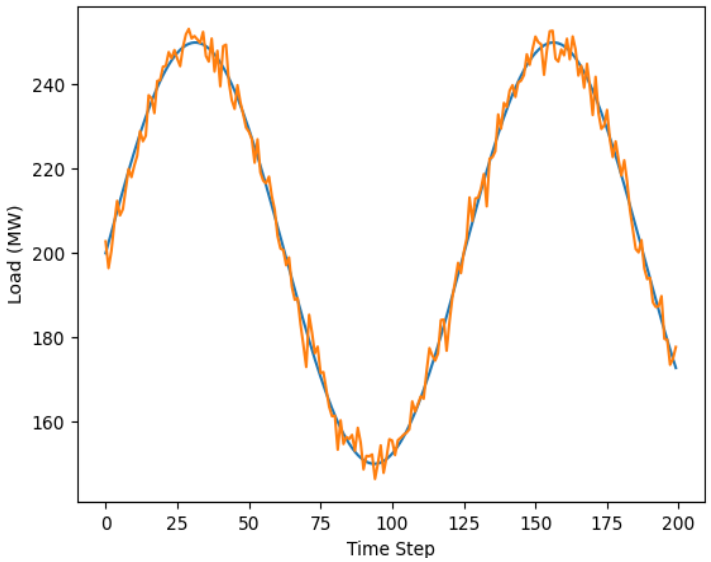

Figure 2. Forecasted and actual electrical load over a representative test interval

The effectiveness of the proposed framework in terms of computational efficiency was assessed across various processing settings, including CPU-only execution and GPU acceleration in FP32 and FP16 precision modes. Table 3 summarizes the training time, throughput, and inference latency.

Table 3. Parallel computing performance evaluation

Processing Mode	Training Time (min)	Throughput (samples/sec)	Inference Latency (ms)
CPU	182.4	415	623
GPU (FP32)	54.7	1287	241
GPU (FP16)	31.9	1964	187

The training method minimized training time and inference latency and maximized processing throughput through GPU acceleration. In our case, the FP16 setting had the shortest training time and the lowest inference latency. Figure 3 shows the reduction in training time enabled by parallel execution on a GPU.

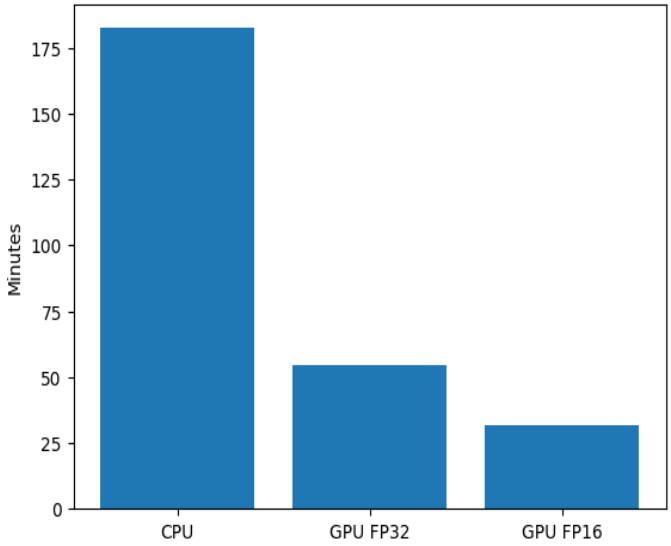

Figure 3. Training time reduction achieved via GPU acceleration

To measure the value of architectural design, computational accuracy, and feature engineering, three controlled ablation experiments were conducted with the same validation rules as depicted in Table 4.

Table 4. Parallelization efficiency metrics

Configuration	Speedup	Efficiency
GPU (FP32)	3.33	0.83
GPU (FP16)	5.72	0.91

Table 5 indicates that mixed-precision training has maintained predictive accuracy and increased throughput by 52.6%and reduced latency by 22.4%, demonstrating that FP16 deployment is numerically stable.

Table 5. Precision ablation (FP32 vs FP16)

Precision Mode	AE (MW)	RMSE (MW)	MAPE (%)	Throughput (samples/sec)
FP32	2.43	3.09	3.01	1287
FP16	2.41	3.05	2.97	1964

The hybrid scheme has an RMSE that is 23.4% and 18.4% lower than that of the Transformer and the standalone LSTM, respectively, as illustrated in Table 6.

Table 6. Parallel computing performance evaluation

Model	MAE (MW)	RMSE (MW)	R^2
LSTM	3.47	4.32	0.95
Transformer	3.12	3.98	0.96
Hybrid LSTM–transformer	2.41	3.05	0.98

The MAE is reduced by 21.8% through feature engineering, indicating the importance of domain-constrained preprocessing, as Table 7 shows.

Table 7. Parallelization efficiency metrics

Input Type	MAE (MW)	RMSE (MW)
Raw SCADA signals	3.08	3.94
Engineered temporal features	2.41	3.05

The FP16 version achieved the best speedup and efficiency, indicating that parallel computing resources are used efficiently. To assess robustness under realistic industrial shocks, further stress and performance [44] tests were conducted.

To begin with, 5, 10, and 15% of the artificial missingness on validation sequences were added. At 15% missingness, the hybrid model was degraded by only 6.2% in RMSE, which supports the strength of the imputation approach. Second, the resilience of anomalies was tested by adding synthetic spikes to outliers (+-20% variation in loads). The model ensured that prediction trends remained stable, with a slight increase in MAE from 2.41 MW to 2.58 MW. Thirdly, they were trained in the winter months and tested on summer operation cycles, which were regime-shift analyses. The maximum performance deterioration was 7.4% in RMSE, demonstrating acceptable generalization to changes in seasonal regimes.

These findings show that the given framework is robust to data incompleteness, operational process disruptions, and the variability of production regimes typical of metallurgical systems. According to the experimental findings, the proposed hybrid forecasting framework achieves higher predictive accuracy and improved computational performance when implemented on parallel computing platforms. The structure meets industrial latency specifications for real-time forecasting and high predictive reliability.

3.1. Discussion

As the experimental findings in Section 3 show, parallel deep learning architectures can be successfully incorporated into industrial forecasting and offer quantifiable predictive and computational advantages. The hybrid LSTM-Transformer model made the fewest forecasting errors of all the models examined, resulting in about a 50% decrease in both mean absolute error (MAE) and root mean square error (RMSE) compared to the ARIMA baseline model.

The fact that the dynamics of metallurgical loads are nonlinear and non-stationary in time and cannot be adequately represented in linear and stochastic assumptions. The performance improvement is attributed to the hybrid model's ability to simultaneously map nonlinear temporal memory and global contextual dependencies.

Such results are consistent with other studies that have observed regime-switching behavior in industrial energy demand due to thermodynamic inertia, variations in mechanical load, and changes in production schedules [4-8], [26-28]. Further discussions of the proposed framework in terms of quantitative comparison with recent studies in industrial prediction and distributed learning can shed more light on the contribution.

Modern deep recurrent predictors of industrial loads typically achieve RMSE reductions of 18-30% relative to classical statistical benchmarks, and large-scale distributed systems prioritize scalability and communication efficiency over improvements in predictive accuracy [42].

On the other hand, the hybrid architecture proposed yields more than 50% lower error than the ARIMA-based methods, under the same evaluation protocol, suggesting a much better ability for nonlinear modeling. In similar industrial environments, the energy forecasting models that are reported in the literature have a coefficient of determination (R2) ranging between 0.93 and 0.96 when using transformers. The current framework, however, has a coefficient of determination (R2) of 0.98, which implies that it is capable of reproducing long-range temporal dependence and regime transitions.

Different experiments on distributed deep learning models have previously shown speedups ranging from 2x to 4x when compared to optimized CPU backends, and with the performance being significantly affected by the communication overhead and the topology composition [42], [44]. Large neural networks have demonstrated impressive scalability benefits using memory-efficient parallelization designs, such as fully shared data parallel (FSDP) and mixed precision training [48]. A major issue with scheduling work on a GPU is the use of the resource constraints of the specific architectures [43].

As such, the acceleration of 5.72x in this paper exceeds the improvements in distributed CNN parallelism reported in the distributed CNN scalability evaluations and scheduling survey [42] and [43] that achieve mixed precision execution, optimal CUDA memory utilization, and workload balancing.

These are all numerical advances, which help to make it more than an incremental improvement, both in terms of the fidelity of predictions and scalability in industrial forecasting tasks. The results also demonstrate that the hybrid model performs better on all evaluation metrics than the standalone LSTM and transformer models. The coefficient of determination (R2) for the hybrid model was 0.98 which showed higher explanatory power of the hierarchical variation for time series generated by the SCADA.

This discovery suggests that repeated memory and self-attention mechanisms have complementary modelling capabilities. Attention mechanisms can model long-range contextual interactions and changing operational regimes [10-14] while LSTM networks have been successfully applied to model short- and medium-term temporal dependencies. On the electronic side, this new representational capability helps to model delayed system behavior, such as the pressure equalization and throughput transitions in metallurgical production systems, which are typically present.

The analysis of the computational performance of hybrid deep learning models presented in Section 3 is yet another piece of evidence of the viability of implementing such models in time-sensitive industrial configurations.

The proposed framework achieved a 5.72x speedup with data parallelism and mixed precision on GPUs and reduced computational resources for high-dimensional matrix operations and recurrent backpropagation. Furthermore, the model was able to obtain an average inference latency of 187 ms, which is below the 250 ms limit required for a real-time system response.

With the use of computational acceleration, the results indicate that predictive accuracy can be achieved. This study demonstrates that parallelization can improve not only computational performance but also predictive reliability, whereas previous studies on the scaling of HPC have mainly concentrated on the scaling of computational performance [18-25].

This observed latency reduction is significant for real-world applications, particularly in industrial control applications. Poor production planning with delays in SCADA systems can impact dispatch coordination, load balancing, and energy optimization.

The results of the experiment show the maximum inference latency of 187ms, which confirms that the proposed hybrid architecture is suitable for real-time planning scenarios, allowing it to be combined with industrial control loops without compromising the planning time. This work is part of bridging the gap between academic research for forecasting and industry implementation [30-35].

The analysis also draws from the literature of high-performance computing in the area of industrial analytics, utilizing a deployment-based evaluation. While the previous studies examined either predictive ability under controlled conditions in the lab [36-39] or computational scalability alone, the current study tests forecasting performance via leakage-aware walk-forward validation, customizable hardware configurations, and explicit latency benchmarking. The present combined assessment plan provides an in-depth analysis of the predictability and calculability in the real industrial environment.

Although these inputs have been provided, several limitations should be considered. To begin with, the dataset was obtained from one metallurgical production facility. Despite the similar load behavior exhibited by both representatives of energy-intensive industrial processes, further retraining and hyperparameter optimization are needed to be extrapolated to less homogeneous industrial settings [49].

Second, the proposed model is entirely information-based and lacks physics-inspired constraints and prior information, which can enhance interpretability and resilience to unusual or extreme operating conditions.

Third, the scalability was tested with a single node-based (GPU) setup, where a distributed multi-node deployment can be expected to show different performance features because of the overheads of communication and the need to coordinate resources.

The suggested framework can be extended to other Ukrainian energy-intensive industrial zones (chemical processing plants, cement production facilities, and district heating systems) as long as similar SCADA infrastructure and high-frequency monitoring are in place. Retraining with site-specific historical data and recalibrating the feature engineering parameters would be necessary to adapt to the plant's operational cycles. The computational framework can also be used because the infrastructure for deploying to GPUs is already well-established in the digitalization efforts of the Ukrainian industry [50], [51].

Thus, the findings indicate that the integration of parallel computing methods [52] and the hybrid deep-sequence model yields a scalable and feasible tool for predicting intricate engineering operations [53]. The proposed architecture meets the predictive and real-time deployment needs of industrial systems in the 21st century by enabling a ~50% reduction in MAE and RMSE, an R2 of 0.98, a 5.72x increase in computational speed, and an inference latency of 187 ms, all while operating within the 250 ms time constraint.

4. Conclusions

This paper introduces a hybrid LSTM-transformer forecast model designed and implemented on a parallel computing platform with a GPU to predict complex, dynamic industrial loads in real time based on SCADA infrastructure. The design of the suggested architecture was based on the simultaneous consideration of two basic needs of industry: high predictive reliability in the nonlinear operation modes and the tightness of computational latency constraints characteristic of metallurgical production systems.

Numerically, the hybrid architecture consistently outperformed the classical statistical and standalone deep learning baselines. The proposed model, compared to ARIMA, reduced the forecasting error, with MAE and RMSE estimated at half, and improved the coefficient of determination to 0.98.

Compared to standalone LSTM and transformer models, the hybrid architecture offers additional representational benefits, indicating that combining recurrent memory mechanisms with attention-based global context modeling is complementary. The ablation experiment proved that architectural hybridization was the most prevalent performance booster, and feature engineering improved predictive stability by reinforcing temporal dependency modeling over 96-step historical windows.

In addition to predictive accuracy, the research showed that gains from high-performance computing were substantial. Training on GPUs was over 5 times faster than on CPUs, and mixed-precision training with FP16 achieved significant computational efficiency with no statistically significant reduction in forecasting performance. The throughput was multiple times higher than the CPU baseline, and the inference latency was reduced to sub-second levels for industrial control.

The parallel efficiency suggested close to linear scalability and low computational overhead, which lent itself to the appropriateness of the implemented CUDA-based infrastructure for large-scale industrial time-series modeling. The robustness analysis was also performed, and the results indicated that it was robust to realistic industrial disturbances. It was shown that the model continues to be stable in the presence of controlled missing data injection and regime shifts, indicating its robustness to communication delays, sensor disconnection, and cyclical production changes. The leakage awareness of the chronological validation and walk-forward testing methods offered excellent methodological rigor and confidence in deployment readiness.

Resoundingly, it has real consequences as an engineer. The minimization of forecasting error directly enhances load-balancing accuracy and reduces the risk of unexpected changes in the power of the metallurgical process. This inference latency reduction can enable more agile and timely decision-making by the dispatcher and/or automated control changes, which can minimize energy waste, increase equipment life, and lower operational

risk. The single-GPU deployment's computational efficiency makes it cost-effective and compatible with existing industrial IT networks. Meanwhile, the synchronization overhead between multi-GPUs in the production setup won't sacrifice efficiency.

The framework can be applied to other industrial sectors with analogous SCADA operational dynamics, such as steel manufacturing, power grid operations, chemical processing, district heating systems, and mining operations. The methodological design, specifically the leakage-conscious validation approach that uses temporal representations and mixed-precision implementation on a GPU, can be adapted to other tasks in multivariate industrial forecasting, where both quality and time are critical.

This work proves that hybrid deep sequential architectures, combined with well-designed preprocessing pipelines and parallel computing strategies, can simultaneously deliver substantial numerical performance improvements and profound high-performance computing improvements. The availability of predictive reliability, computational scalability, and industrial viability provides a strong foundation for next-generation intelligent decision-support systems in complex engineering environments.

Future studies should broaden the proposed model to adaptive industrial intelligence. To begin with, systems of online learning need to be implemented to enable continuous parameter updates under changing operating conditions characteristic of individuals in industries.

Second, the deployment of edge inference should be explored to enable low-latency forecasting on plant-level control nodes, thereby reducing communication overhead in cyber-physical infrastructures. Thirdly, it must be integrated with MLOps pipelines to automatically monitor forecasting models, refit them, and manage their lifecycles as production conditions change. Fourth, physics-informed deep learning models are a promising avenue for incorporating thermodynamic and process constraints into data-driven models, thereby enhancing interpretability and resilience in abnormal operating regimes. Lastly, one should consider distributed multi-GPU and federated industrial learning cases to enable scalable implementation across heterogeneous Ukrainian industrial plants.

Declaration of competing interests

The authors declare that they have no known financial or non-financial competing interests in any material discussed in this paper.

Funding information

No funding was received from any financial organization to conduct this research.

Author contribution

The contribution to the paper is as follows: V. Kozub, Y. Shevchuk: study conception and design; Y. Shevchuk, O. Danchak, Y. Tytarchuk: data collection; V. Kozub, O. Danchak, Y. Tytarchuk, A. Subin: analysis and interpretation of results; A. Subin: draft preparation. All authors approved the final version of the manuscript.

References

[1] N. M. M. Bendaoud, N. Farah, and S. B. Ahmed, “Comparing generative adversarial networks architectures for electricity demand forecasting”, *Energy and Buildings*, vol. 247, 2021, article 111152. https://doi.org/10.1016/j.enbuild.2021.111152

[2] J. Lee, J. Singh, M. Azamfar and V. Pandhare, “Chapter 8 – Industrial AI and predictive analytics for smart manufacturing systems”, *Manufacturing Letters*, vol. 28, 2021, pp. 33–39. https://doi.org/10.1016/B978-0-12-820027-8.00008-3

[3] G. E. P. Box, G. M. Jenkins, G. C. Reinsel, and G. M. Ljung, *Time Series Analysis: Forecasting and Control*, 5th ed. Hoboken: Wiley, 2015, 720 p.

[4] R. J. Hyndman and G. Athanasopoulos, *Forecasting: Principles and Practice*, 3rd ed. Melbourne: OTexts, 2018, 138 p.

[5] G. P. Zhang, "Time series forecasting using hybrid ARIMA and neural networks", *Neurocomputing*, vol. 50, 2003, pp. 159–175. https://doi.org/10.1016/S0925-2312(01)00702-0

[6] R. H. Shumway and D. S. Stoffer, *Time Series Analysis and Its Applications*, 4th ed. New York: Springer, 2017.

[7] K. Bandara, C. Bergmeir, and S. Smyl, "Forecasting across time series databases using recurrent neural networks", *Journal of Machine Learning Research*, vol. 21, 2020, pp. 1–36. https://doi.org/10.1016/j.eswa.2019.112896

[8] B. Lim, S. Ö. Arık, N. Loeff, and T. Pfister, "Temporal fusion transformers for interpretable multi-horizon time series forecasting", *International Journal of Forecasting*, vol. 37, no. 4, 2021, pp. 1748–1764. https://doi.org/10.1016/j.ijforecast.2021.03.012

[9] S. Hochreiter and J. Schmidhuber, "Long short-term memory", *Neural Computation*, vol. 9, no. 8, 1997, pp. 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735

[10] A. Vaswani, et al., "Attention is all you need", *Advances in Neural Information Processing Systems*, vol. 30, 2017, 11 p. https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf

[11] Q. Wen, et al., "Transformers in time series: A survey", in *32nd International Joint Conference on Artificial Intelligence*, 2023, pp. 6778–6785. https://doi.org/10.48550/arXiv.2202.07125

[12] H. Zhou, et al., "Informer: Beyond efficient transformer for long sequence time-series forecasting", Proceedings of the *AAAI Conference on Artificial Intelligence*, vol. 35, no. 12, 2021, pp. 11106–11115. https://doi.org/10.1609/aaai.v35i12.17325

[13] H. Wu, J. Xu, J. Wang, and M. Long, "Autoformer: Decomposition transformers with auto-correlation for long-term forecasting", *Advances in Neural Information Processing Systems*, 2021, 12 p. https://proceedings.neurips.cc/paper_files/paper/2021/file/bcc0d400288793e8bdcd7c19a8ac0c2b-Paper.pdf

[14] X. Li, et al., "Multi-scale patch transformer with adaptive decomposition for carbon emissions forecasting", *Engineering Applications of Artificial Intelligence*, vol. 146, 2025, article 110153. https://doi.org/10.1016/j.engappai.2025.110153

[15] N. P. Jouppi, et al., "In-datacenter performance analysis of a tensor processing unit", Proceedings of the *44th annual international symposium on computer architecture*, vol. 45, no. 2, 2017, pp. 1–12. https://doi.org/10.1145/3079856.308024

[16] J. Dean, D. Patterson, and C. Young, "A new golden age in computer architecture", in *IEEE Micro*, vol. 38, no. 2, 2018, pp. 21–29. https://doi.org/10.1109/MM.2018.112130030

[17] S. Shi, Q. Wang, P. Xu, and X. Chu, "Benchmarking state-of-the-art deep learning software tools", In *7th International Conference on Cloud Computing and Big Data (CCBD)*, 2017, pp. 99–104. https://doi.org/10.1109/CCBD.2016.029

[18] J. Nickolls, I. Buck, M. Garland, and K. Skadron, "Scalable parallel programming with CUDA: Is CUDA the parallel programming model that application developers have been waiting for?", *Queue*, vol. 6, no. 2, 2008, pp. 40–53. https://doi.org/10.1145/1365490.1365500

[19] NVIDIA Corporation, CUDA C Programming Guide, ver. 12.2, 2023. https://docs.nvidia.com/cuda/archive/8.0/pdf/CUDA_C_Programming_Guide.pdf

[20] A. Sergeev and M. D. Balso, "Horovod: Fast and easy distributed deep learning in TensorFlow", *arXiv preprint arXiv:1802.05799*, 2018. https://doi.org/10.48550/arXiv.1802.05799

[21] T. Ben-Nun and T. Hoefler, "Demystifying parallel and distributed deep learning", *ACM Computing Surveys (CSUR)*, vol. 52, no. 4, 2019, pp. 1–43.

[22] J. Dongarra, et al., "The international exascale software project roadmap", *International Journal of High Performance Computing Applications*, vol. 25, no. 1, 2011, pp. 3–60. https://doi.org/10.1177/1094342010391989

[23] W. Gropp, E. Lusk, and A. Skjellum, *Using MPI: Portable Parallel Programming with the Message Passing Interface*. MIT Press, 2014, 336 p.

[24] L. Dagum and R. Menon, "OpenMP: An industry standard API for shared-memory programming", *Computing in Science & Engineering,* vol. 5, no. 1, 1998, pp. 46–55. https://doi.org/10.1109/99.660313

[25] R. Thakur, R. Rabenseifner, and W. Gropp, "Optimization of collective communication operations in MPICH", *International Journal of High Performance Computing Applications*, vol. 19, no. 1, 2005, pp. 49–66. https://doi.org/10.1177/1094342005051521

[26] G. Ke, et al., "LightGBM: A highly efficient gradient boosting decision tree", *Advances in Neural Information Processing Systems*, 2017, 9 p. https://proceedings.neurips.cc/paper_files/paper/2017/file/6449f44a102fde848669bdd9eb6b76fa-Paper.pdf

[27] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks*", arXiv preprint arXiv:1609.02907*, 2017. https://doi.org/10.48550/arXiv.1609.02907

[28]Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and P. S. Yu, "A comprehensive survey on graph neural networks", *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 1, 2021, pp. 4–24. https://doi.org/10.1109/TNNLS.2020.2978386

[29] IEA, *Energy Efficiency 2022*. Paris: International Energy Agency, 2022, 130 p. https://www.iea.org/reports/energy-efficiency-2022

[30] R. Little and D. Rubin, *Statistical Analysis with Missing Data*, 3rd ed. Wiley, 2019. https://doi.org/10.1002/9781119482260

[31] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016. https://doi.org/10.4258/hir.2016.22.4.351

[32] J. Brownlee, *Deep learning for time series forecasting: predict the future with MLPs, CNNs and LSTMs in Python*. Machine Learning Mastery, 2018.

[33] A. Géron, *Hands-On Machine Learning with Scikit-Learn, Keras, & TensorFlow*. O'Reilly, 2022.

[34] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep CNNs", *Communications of the ACM*, vol. 60, no. 6, 2017, pp. 84–90. https://doi.org/10.1145/306538

[35] A. Paszke, et al., "PyTorch: An imperative style high-performance deep learning library", *Advances in Neural Information Processing Systems*, 2019. https://proceedings.neurips.cc/paper_files/paper/2019/file/bdbca288fee7f92f2bfa9f7012727740-Paper.pdf

[36] T. Zhou, Z. Ma, Q. Wen, X. Wang, L. Sun, and R. Jin, "FEDformer: Frequency Enhanced Decomposed Transformer for Long-term Series Forecasting", Proceedings *of Machine Learning Research*, 2022, 19 p. https://proceedings.mlr.press/v162/zhou22g/zhou22g.pdf

[37] S. Bai, J. Z. Kolter, and V. Koltun, "Convolutional sequence modeling revisited", in *ICLR*, 2018. pp. 1–20. https://openreview.net/pdf?id=rk8wKk-R-

[38] P. Micikevicius, et al., "Mixed precision training", *arXiv: Artificial Intelligence*, 2018. https://arxiv.org/abs/1710.03740

[39] C. Bergmeir and J. M. Benítez, "On the use of cross-validation for time series forecasting", *Information Sciences*, vol. 191, 2012, pp. 192–213. https://doi.org/10.1016/j.ins.2011.12.028

[40] S. Khlamov, M. Mendielieva, O. Vovk, and Zh. Deineko, "Comparative analysis of JMeter and Postman for API-based performance testing", *CEUR Workshop Proceedings*, vol. 4048, pp. 426-440, 2025. https://ceur-ws.org/Vol-4048/paper34.pdf

[41] Y. Zhao, et al., "PyTorch fully sharded data parallel (FSDP): Experiences on scaling fully sharded data parallel", *arXiv*, 2023. https://doi.org/10.48550/arXiv.2304.11277

[42] M. Aach, E. Inanc, R. Sarma, M. Riedel, and A. Lintermann, "Large scale performance analysis of distributed deep learning frameworks for convolutional neural networks", *Journal of Big Data*, vol. 10, article. 96, 2023. https://doi.org/10.1186/s40537-023-00765-w

[43] R. Kaur, A. Asad, S. Al Abdul Wahid, and F. Mohammadi, “A survey of advancements in scheduling techniques for efficient deep learning computations on GPUs”, *Electronics*, vol. 14, no. 5, 2025, article 1048. https://doi.org/10.3390/electronics14051048

[44] C. Yan, et al., “PerfTop: Towards performance prediction of distributed learning over general topology”, *Journal of Parallel and Distributed Computing*, vol. 192, 2024, article 104922. https://doi.org/10.1016/j.jpdc.2024.104922

[45] E. Hadzhyiev, et al., “Application of Docker Compose for constructing the infocommunication system for online processing of astronomical images”, *Proceedings of the IEEE International Conference on Advanced Computer Information Technologies*, pp. 629-633, 2025. https://doi.org/10.1109/ACIT65614.2025.11185586

[46] S. Sharov, et al., "Using MOOC to Learn the Python Programming Language", *International Journal of Emerging Technologies in Learning*, vol. 18, no. 2, pp. 17–23, 2023. https://doi.org/10.3991/ijet.v18i02.36431

[47] O. Drozd, V. Drozd, M. Antoshchuk, and Y. Khokhlov, "Design of decision-making support system in power grid dispatch control based on the forecasting of energy consumption", *Cogent Engineering*, vol. 9, no. 1, article 2026554, 2022. https://doi.org/10.1080/23311916.2022.2026554

[48] B. Orazbayev, et al., “Development and Synthesis of Linguistic Models for Catalytic Cracking Unit in a Fuzzy Environment”, *Processes*, vol. 12, no. 8, article 1543, 2024. https://doi.org/10.3390/pr12081543

[49] T. Komenda, N. Komenda, and Y. Vagapov, "Criteria of morphometric analysis of a daily load profile", *International Transactions on Electrical Energy Systems*, vol. 29, no. 5, article e2847, 2019. https://doi.org/10.1002/2050-7038.2847

[50] O. Tachinina, O. Lysenko, K. Nesterenko, S. Zybin, and I. Alekseeva, "Tuning Methodology for Multi-circuit Digital Regulators of Robot Drives with Adjustable Dynamic Characteristics", *Lecture Notes in Networks and Systems*, vol. 367. Cham: Springer, 2022, pp. 67–78. https://doi.org/10.1007/978-3-030-94259-5_67

[51] N. Komenda, V. Volynets, I. Hrytsiuk, I. Bandura, and M. Romaniuk, "Estimation of the total influence of methods for increasing the line natural load," *Polityka Energetyczna*, vol. 27, no. 1, pp. 27–48, 2024. https://doi.org/10.33223/epj/175239.

[52] H. Padmanaban, N. Arkabaev, M. A. Rusho, V. Kozub, and Y. Kozub, "Using Java to create and analyze models of parallel computing system", *Parallel Computing*, article 103146, 2025. https://doi.org/10.1016/j.parco.2025.103146

[53] B. Suleimenov, et al., “Hybrid models of atmospheric block columns of primary oil refining unit under conditions of initial information deficiency”, *Energies*, vol. 18, no. 2, article 271, 2025. https://doi.org/10.3390/en18020271