DOI: https://doi.org/10.32782/2524-0072/2026-84-67

УДК 004.8:005.95/.96

FORECASTING EMPLOYEE PRODUCTIVITY USING LSTM NETWORKS: SYNTHETIC DATA APPROACH AND BASELINE COMPARISON

ПРОГНОЗУВАННЯ ПРОДУКТИВНОСТІ ПРАЦІВНИКІВ НА ОСНОВІ LSTM-МЕРЕЖ: СИНТЕТИЧНЕ МОДЕЛЮВАННЯ ТА ОЦІНКА ТОЧНОСТІ

Yakymiv Andrii
PhD Student,
Kyiv National University of Construction and Architecture
ORCID: https://orcid.org/0009-0000-1115-2905

Yehorchenkova Nataliia
Doctor of Technical Sciences, Professor,
Kyiv National University of Construction and Architecture
ORCID: https://orcid.org/0000-0001-5970-0958

Якимів Андрій Романович, Єгорченкова Наталія Юріївна
Київський національний університет будівництва і архітектури

Forecasting employee productivity is becoming an important task in modern project management environments characterized by high dynamics and digital transformation. The aim of this study is to develop and investigate an approach for predicting employee productivity using Long Short-Term Memory (LSTM) neural networks. The methodology is based on modeling temporal sequences of HR metrics describing employee activity and performance in project teams. The results demonstrate that LSTM models can effectively capture temporal dependencies in productivity indicators and provide accurate forecasts of performance dynamics. The practical value of the study lies in the possibility of integrating such models into HR analytics and project management systems to support performance monitoring and managerial decision-making. In addition, the proposed approach can contribute to improving resource planning and early identification of changes in team performance.

Keywords: employee productivity, neural networks, LSTM, IT project management, performance evaluation.

У сучасних умовах цифровізації організацій та зростання складності управління IT-проєктами особливої актуальності набуває застосування інтелектуальних методів аналізу даних для оцінювання та прогнозування продуктивності працівників. Традиційні підходи до аналізу ефективності діяльності персоналу переважно базуються на статичних показниках або періодичних експертних оцінках і часто не враховують часову динаміку змін продуктивності працівників у процесі виконання проєктних завдань. Це обмежує можливості своєчасного виявлення тенденцій у зміні ефективності роботи команд та ускладнює прийняття управлінських рішень. Метою дослідження є розроблення та дослідження підходу до прогнозування продуктивності працівників на основі рекурентних нейронних мереж типу LSTM, здатних моделювати часові залежності у даних про діяльність працівників у проєктному середовищі. У роботі розглядається використання часових рядів HR-метрик, які характеризують різні аспекти індивідуальної та командної ефективності, а також досліджується можливість застосування нейронних мереж для виявлення прихованих закономірностей у зміні показників продуктивності. Методика дослідження передбачає побудову нейронної моделі прогнозування, формування послідовностей вхідних даних на основі часових рядів показників діяльності працівників, а також застосування процедур попередньої обробки даних, зокрема нормалізації показників і формування навчальних вибірок. Для перевірки ефективності запропонованого підходу проведено експериментальне моделювання, що дозволило оцінити здатність нейронної мережі відтворювати часові залежності у зміні продуктивності та формувати прогнози її подальшої динаміки. Отримані результати показали, що використання LSTM-мереж дозволяє ефективно враховувати часову структуру даних та забезпечує високу точність прогнозування змін продуктивності працівників. Практична цінність дослідження полягає у можливості використання запропонованого підходу в системах HR-аналітики та управління проєктами для автоматизованого аналізу продуктивності команд, підтримки прийняття управлінських рішень та підвищення ефективності використання трудових ресурсів.

Ключові слова: продуктивність працівників, нейронні мережі, LSTM, управління IT-проєктами, оцінка ефективності.

Statement of the problem. In modern project management, especially in the field of information technology, there is a growing need for accurate, adaptive and personalized approaches to assessing and predicting employee productivity. The high dynamics of work processes, flexible development methodologies, the spread of remote and hybrid work formats – all this complicates the use of classic assessment methods, such as KPIs, expert scales or even linear regression. These methods mostly do not take into account the time dependence of employee behavior and are prone to losing the context of previous periods.

Previous studies have proposed alternative models, including regression analysis and Bayesian networks, that can improve estimation accuracy through statistical approaches, causal analysis, or probabilistic modeling. However, these approaches also have limitations in capturing complex nonlinear relationships between variables and the dynamics of performance changes over time.

One of the promising directions for solving this problem is the use of recurrent neural networks of the LSTM (Long Short-Term Memory) type, which are able to store information about previous states and adaptively take into account time dependencies. LSTM models have already shown high efficiency in time series forecasting tasks, in particular in economics, medicine, and financial analysis. Their ability to "remember" key events in an employee's history (load, burnout, role changes, project phases) makes them relevant for productivity tasks in project management.

On the other hand, it should be noted that building an effective LSTM model requires a significant amount of high-quality historical data on employee behavior, their participation in project activities, workload changes, etc. Thus, this approach is most suitable for medium and large companies that have the ability to systematically store and analyze HR metrics and meta-information about teamwork.

The aim of this work is to develop and study a model for predicting employee productivity based on LSTM neural networks, built on data about previous periods of work of employees in IT teams. The article considers the architecture of the model, the principles of training input sequences, as well as the results of its testing and comparison with baseline approaches.

Analysis of recent research and publications. LSTM for time series prediction. In modern research, LSTM networks are actively used for time series forecasting – from economic statistics to production KPI control. For example, LSTM models have demonstrated a significant advantage over traditional ARIMA approaches, reducing the forecast error by 84–87% [1]. This is due to the ability of LSTM to take into account both short-term and long-term dependencies in the data, which is not available for classical statistical models. In addition, LSTMs cope well with seasonality, nonlinearity, and noise, which are often encountered in applied time series.

LSTM in industrial applications. In the manufacturing sector, LSTMs have been used to predict key performance indicators (KPIs) of manufacturing systems. In particular, research has shown that LSTM networks are able to account for temporal dependencies and achieve stable forecast accuracy across different configurations [2]. Maier & Wrobel demonstrated that even under conditions of changes in production schedules or logistics, LSTM models remain robust to retraining and maintain forecast quality. This approach opens up the possibility of flexible resource management and early detection of bottlenecks in the production cycle.

Application of LSTM in HR analytics. In HR analytics, although there are fewer examples, LSTMs have been used to predict employee turnover and identify churn patterns. In particular, combined RNN-HMM-LSTM models have been used to analyze employee turnover characteristics [3]. In a study by Shi et al., it was found that such a hybrid architecture allows not only to predict the probability of turnover, but also to detect hidden patterns in employee behavior. This is especially important in large organizations, where traditional analysis cannot keep up with the volume of changes and interactions.

Optimization of HR processes via LSTM. In addition, the Balanced HR allocation approach based on the LSTM framework was presented at the international conference, which combined normalization, PCA, and deep learning for optimal resource allocation, demonstrating an accuracy of over 95% [4]. The proposed system allows predicting team effectiveness depending on the current workload distribution and recommended changes. The model also integrates information about project cycles, adaptation periods, and role changes, which is critically important in a dynamic project environment.

Overall, LSTM has proven its effectiveness in complex time-varying forecasting tasks, both in industry and HR analytics. However, no full-fledged solutions for predicting employee

productivity in the context of HR metrics have been proposed yet. This creates a clear scientific niche that allows us to propose a forecasting model based on HR behavioral series focused on productivity tasks in a team environment

Highlighting previously unresolved parts of the overall problem. Despite the growing interest in applying machine learning and neural network models to the analysis of organizational processes, a number of important issues related to forecasting employee productivity remain insufficiently studied. In particular, most existing approaches to performance evaluation rely on static indicators or aggregated metrics that do not adequately reflect the dynamic nature of employee behavior in project environments.

In modern IT projects, productivity indicators often change over time under the influence of multiple factors such as workload fluctuations, team interactions, learning effects, and periods of peak activity or fatigue. However, many traditional analytical approaches are not capable of effectively capturing temporal dependencies between such factors. This creates a need for methods that can model sequential patterns and long-term dependencies in employee performance data.

Another unresolved aspect concerns the application of deep learning models, particularly recurrent neural networks, in HR analytics and project management research. Although LSTM architectures have proven effective in time-series forecasting tasks in fields such as finance, industry, and healthcare, their use for modeling employee productivity dynamics remains relatively limited. In particular, there is a lack of studies that systematically evaluate the effectiveness of such models in comparison with simpler baseline forecasting approaches.

In addition, practical challenges arise from the limited availability of structured historical HR data in many organizations. This creates methodological difficulties in training deep learning models and requires the development of alternative approaches for modeling and validating forecasting systems. Therefore, further research is needed to investigate how temporal neural network models can be applied to productivity forecasting tasks and how their predictive performance compares with traditional analytical methods

Formation of the objectives of the article (task statement). The aim of this study is to develop and investigate an approach for forecasting employee productivity using LSTM neural networks in the context of project management. The research focuses on modeling temporal dependencies in HR-related performance metrics and evaluating the predictive capability of recurrent neural network architectures for analyzing employee productivity dynamics. Particular attention is given to constructing and testing an LSTM-based model using time-series data that reflect key indicators of employee activity and effectiveness. The study also includes a comparative assessment of the proposed model with baseline forecasting approaches in order to determine its practical applicability for HR analytics and decision support in project management environments.

Summary of the main research material. LSTM (Long Short-Term Memory) is a type of recurrent neural network (RNN) that is specifically designed to avoid the problem of fading gradients when working with long sequences [5; 6]. The key feature of LSTM is the presence of a built-in memory, which allows it to “remember” important information over a large number of time steps. This is achieved through three special “gates”: forget gate, input gate and output gate, which control the flow of information in the memory cell. Each such block takes as input not only the current value of the features, but also the previous hidden state and the internal state of the memory, which allows the model to take into account sequential dependencies in the data. This is especially useful in tasks where the value of a variable at a certain point in time depends on events in the past – for example, in financial analysis, biomedical signals, climate modeling [6].

Input data generation. LSTM models do not work with single examples, but with sequences of fixed length. That is, instead of passing the model individual rows of a table, as in classical regression, we build "time windows" – for example, sequences of length $T = 10$, which represent 10 consecutive observations.

Each such sequence has the form T x K, where:

– T – number of time steps (window length);

– K – the number of features in each step.

The target variable (output) is either the value of one of the features one step ahead (time forecast), or a separate variable that summarizes the state of the system (e.g., failure risk, index, etc.). Forming such windows is called the "sliding window approach" and is a standard practice for preparing data for training temporal models.

Preprocessing: normalization and feature selection. LSTM is a differential model that is sensitive to the scale of the input variables [7].

Therefore, prior normalization of the features is critical. The most common options are:

– Z-score normalization: subtracting the mean and dividing by the standard deviation

– Min-Max scaling: scaling values to the interval [0, 1] [8].

If the number of input features is large (e.g., >30), dimensionality reduction methods such as PCA (Principal Component Analysis) [9] may be used. Filtering or object-oriented feature selection methods can also be used, for example, based on correlation with the target variable or feature importance from Random Forest-type models [10].

Neural network architecture. A typical LSTM network consists of the following components [11]. The *input layer*, which accepts a three-dimensional tensor (batch_size, time_steps, features). Then it has 1 or 2 *LSTM layers* with hyperparameters: number of units (e.g., 64 or 128), initialization type, return sequences. It has *dropout layer* to combat overfitting and *dense layers* (fully connected) with the number of outputs corresponding to the type of problem (Linear activation or Softmax activation).

Batch Normalization can also be applied, which stabilizes gradients and speeds up learning [12]. Training and validation. To train an LSTM model, one of the typical loss functions is used [13]:

– MSE (mean squared error) – for regression problems;

– MAE (mean absolute error) is less sensitive to outliers;

– Categorical cross-entropy – for multi-class classification.

Optimization is performed using the gradient descent algorithm (most often Adam), using Backpropagation Through Time (BPTT) [14].

Training samples are built based on chronological data division – training, validation, and test periods should not overlap in time. This ensures that the model does not see information from the future. Early stopping is also often used – a mechanism for stopping training when the validation error stops decreasing.

Indicators selection. Since this study did not have access to large amounts of real HR data, it was decided to generate a synthetic dataset that was as close in structure and statistical properties to possible real-world employee productivity indicators as possible.

Selection of indicators (features). Four key quantitative characteristics, often used in project management and HR analytics systems, were selected for modeling:

– X_1 – Number of completed tasks (tasks_completed) – the main indicator of productivity.

– X_2 – Estimate Accuracy (%) (estimate_accuracy) – the difference between the planned and actual task completion time.

– X_3 – Number of defects found (bugs_found) – an indicator of the quality of work.

– X_4 – Review activity (code_reviews) – participation in team processes.

The target variable Y is the integral productivity index (0–100).

Synthetic data generation. To reflect the characteristics of real data, the following principles were taken into account:

– *Limited range of most values:* for example, the number of tasks per week is on average 7–10, but with occasional spikes to 15 or drops to 3 (simulating peak and crisis weeks).

– *Unpredictability (noise):* Random fluctuations were added to the baseline values to simulate the impact of factors that cannot be directly measured (stress, project changes, vacation, etc.).

– *Rare anomalies:* periodically, values that differed significantly from the average appeared in the data (for example, 20 completed tasks or zero review participation) so that the model could learn to take them into account.

Thus, we modeled data with a relatively stable "core" and random deviations that better reflect real-world workflows.

Target formation and Normalization. The integral performance Y was calculated as a weighted sum of the four features with the addition of random noise to simulate the subjectivity of the estimates:

$$Y = w_1 \cdot X_1 + w_2 \cdot X_2 - w_3 \cdot X_3 + w_4 \cdot X_4 + \varepsilon \qquad (1)$$

where the coefficients w_i specified the contribution of each feature (for example, the largest weight for the number of tasks completed), and a ε is the random error. Since LSTMs are sensitive to feature scales, all input data were normalized to the range [0,1] using Min-Max Scaling. This allowed for faster model convergence and reduced the risk of favoring features with larger scales.

The target variable was also normalized and, after prediction, scaled back to the original range for interpretation of the results.

LSTM works with sequential data windows. For each forecast point, a window of T previous periods (e.g., 12 weeks) was formed, and the target value was the performance in the next period. This allowed the model to take into account temporal dependencies and trends.

Model architecture and settings. After data preparation and normalization, a model was built based on Long Short-Term Memory (LSTM), a type of recurrent neural network (RNN), which is well suited for time series modeling due to its ability to store information about previous states over long intervals.

Reasons for choosing LSTM

– Modeling time dependencies – employee productivity depends not only on current indicators, but also on trends from previous periods (for example, accumulation of fatigue or increased efficiency after training).

– Protection against “gradient fading” – in classical RNNs it is difficult to take into account long sequences, but the “forget gate” mechanism in LSTM allows you to retain important information and discard irrelevant information.

– Flexibility in working with noisy data – thanks to the built-in mechanism for selecting important features in the sequence, LSTMs are less sensitive to random fluctuations in the data, which is critical for HR metrics.

Three main metrics were used to evaluate the effectiveness of the LSTM model:

– MSE (Mean Squared Error) is a mean square error that shows the average square of the deviation of the forecast from the fact. It is sensitive to large errors, so it is a good indicator of serious mistakes.

– MAE (Mean Absolute Error) is the average absolute error, convenient for interpretation in the original scale (in our case, in performance scores).

– R^2 (Coefficient of Determination) – the proportion of data variance explained by the model (the closer to 1, the better the forecast; negative values indicate that the model is worse than the naive strategy).

For a correct comparison, the LSTM results were compared with two baseline models:

– Mean base model. The prediction at each step is equal to the average value of the target variable in the training set:

$$\hat{y}_t = \frac{1}{N}\sum_{i=1}^{N} y_i \qquad (2)$$

Such a model does not take into account the temporal structure of the data and is a "minimum threshold" – if the model does not exceed its accuracy, it has no practical meaning [15].

– Persistence model. The forecast of the next value is equal to the previous observation:

$$\hat{y}_t = y_{t-1} \qquad (3)$$

This approach is effective for data with strong autocorrelation, where values change gradually. Persistence is a more complex benchmark than Mean base because it takes into account time dependence [16]. The same metrics as for LSTM (MSE, MAE, R^2) were calculated for both models to assess how much the proposed approach outperforms the baseline solutions.

Model evolution and results.

Stage 1 – Initial forecast: «straight line»:

In the first iteration, the model generated a practically constant forecast that did not take into account changes in the data. Reasons were insufficient training set (several hundred examples) with weak architecture (1 LSTM layer with a small number of units). Also we defined synthetic data without clear time dependencies.

Table 1

Stage 1 models comparison results

	MSE	MAE	R^2
LSTM	83.10	6.92	0.00
Mean base	-	-	0.00

Source: compiled by the author based on own research and computational experiments

Stage 2 – Increasing data and number of epochs:

After increasing the training set to 5000 examples and adding a second LSTM layer, the prediction became more sensitive to general trends, but still did not reproduce local fluctuations. The model began to “catch up” with the average performance level, but smoothed out the peak values.

Table 2

Stage 2 models comparison results

	MSE	MAE	R^2
LSTM	437.41	20.28	-9.02
Mean base	43.67	5.13	0.00

Source: compiled by the author based on own research and computational experiments R^2 = –9.02. The result is strongly negative, meaning that the LSTM predicts significantly worse than a simple “take the average” strategy. This indicates problems with the architecture and data matching

Stage 3 – Introducing dependencies into the data:

We modified the synthetic data generator:

– added lag dependencies (t – 2, t – 4, t – 6);

– introduced a correlation between features (for example, X_3 the number of bugs began to have a stronger influence on (Y);

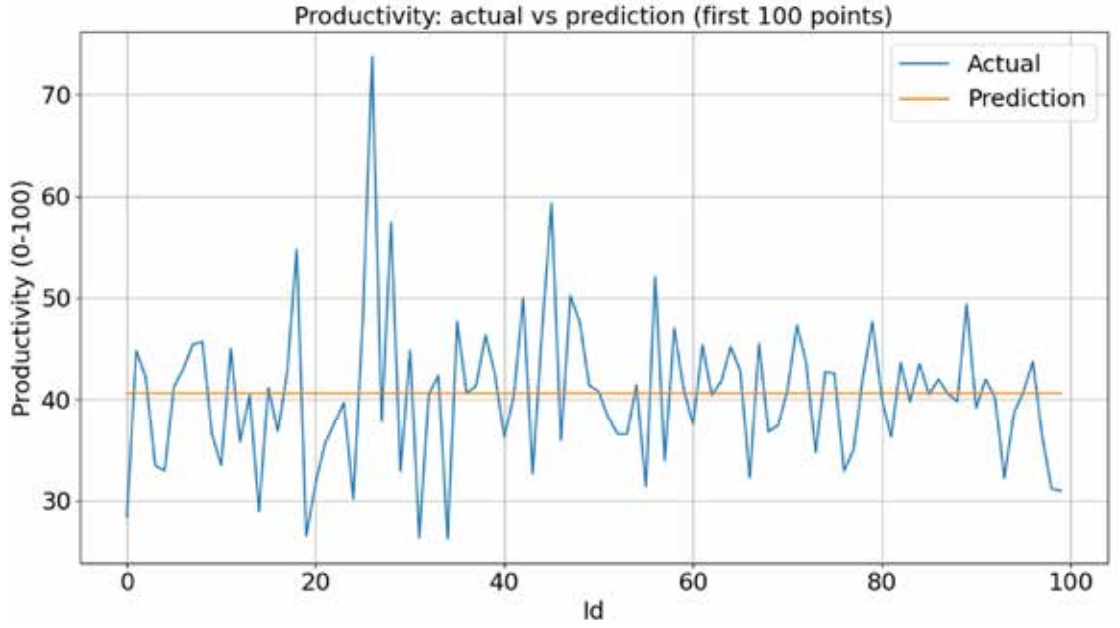

Figure 1. Initial forecast, the model ignores fluctuations in the data

Source: compiled by the author based on own research and computational experiments

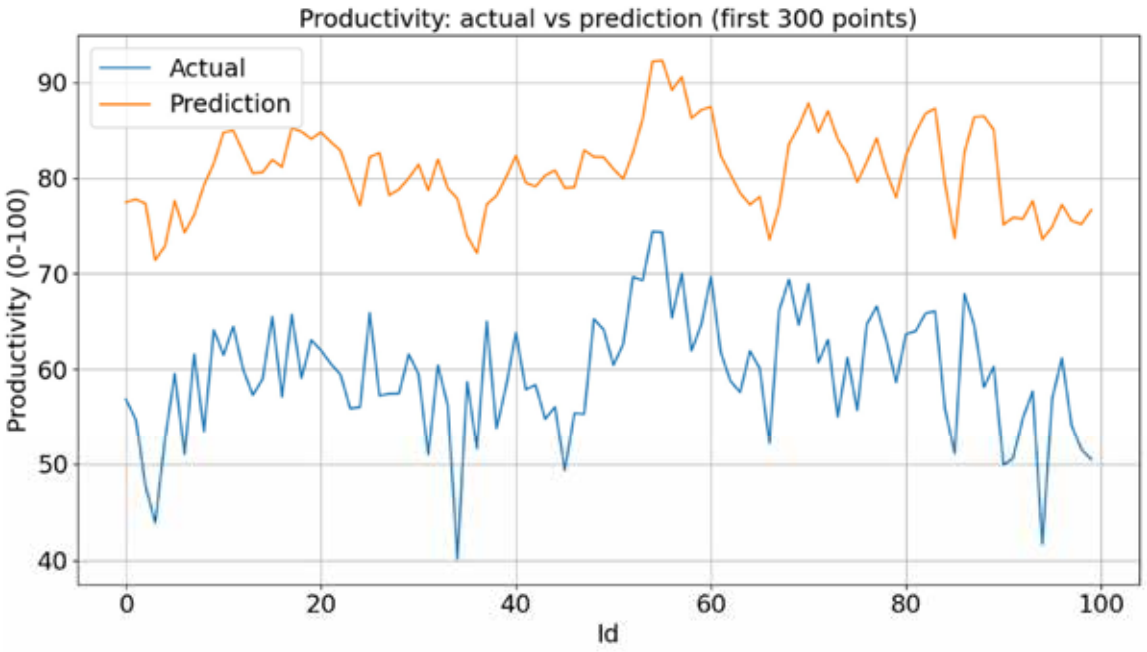

FigURE 2. Forecast after data augmentation – the curve is responsive, but too smooth

Source: compiled by the author based on own research and computational experiments

– reduced the first-order autocorrelation (t – 1) to reduce the advantage of the persistence baseline.

These changes forced the model to look for more complex patterns and significantly improved its ability to reproduce actual fluctuations.

Although persistence works R^2 = –0.62 (quite well), LSTM still lags behind due to insufficient training on the new data structure.

Stage 4 – Final architecture optimization:

We selected the optimal window length time-steps, configured regularization (Dropout = 0.3), introduced the EarlyStopping and

ReduceLROnPlateau mechanisms to optimize learning speed. We receive a result where the model reproduces both long-term and short-term fluctuations with high accuracy ($R^2 \approx 0.95$).

Table 3

Stage 3 models comparison results

	MSE	MAE	R^2
LSTM	124.09	10.39	-1.62
Mean base	47.41	5.54	0.00
Persistence	17.95	3.32	0.62

Source: compiled by the author based on own research and computational experiments

Table 4

Stage 4 models comparison results. Source: compiled by the author based on own research and computational experiments

	MSE	MAE	R^2
LSTM	10.47	2.57	0.95
Mean base	190.82	11.07	0.00
Persistence	261.87	13.01	-0.37

Source: compiled by the author based on own research and computational experiments

R^2 = 0.95 the model reproduces 95% of the variations in the data, significantly outperforming both Mean base and Persistence. This shows that LSTM is able to effectively model temporal dependencies in HR metrics. Negative Persistence implies that this prediction model performs worse than just the average.

The dynamics of the decrease in the mean square error in the normalized scale (0–1) during model training is shown. The low and stable level of validation error after ~10th epoch indicates effective model generalization and the absence of overtraining.

The points are located densely along the diagonal, indicating high accuracy of the model (R^2 = 0.95) and the absence of significant systematic biases in the predictions. The developed LSTM model showed high predictive accuracy when applied to synthetically generated productivity data, achieving performance metrics of MSE = 10.47, MAE = 2.57, and R^2 = 0.95 on the test sample. The obtained results significantly outperform the baseline forecasting approaches, including the Mean base and Persistence models, confirming the effectiveness of using deep recurrent neural networks for time-dependent productivity prediction tasks.

Conclusions. The conducted research achieved the stated objective of developing and investigating an approach for forecasting employee productivity using LSTM neural networks in the context of project management. The study demonstrated that recurrent neural network architectures are capable of effectively modeling temporal dependencies in HR metrics and capturing complex patterns in employee performance dynamics.

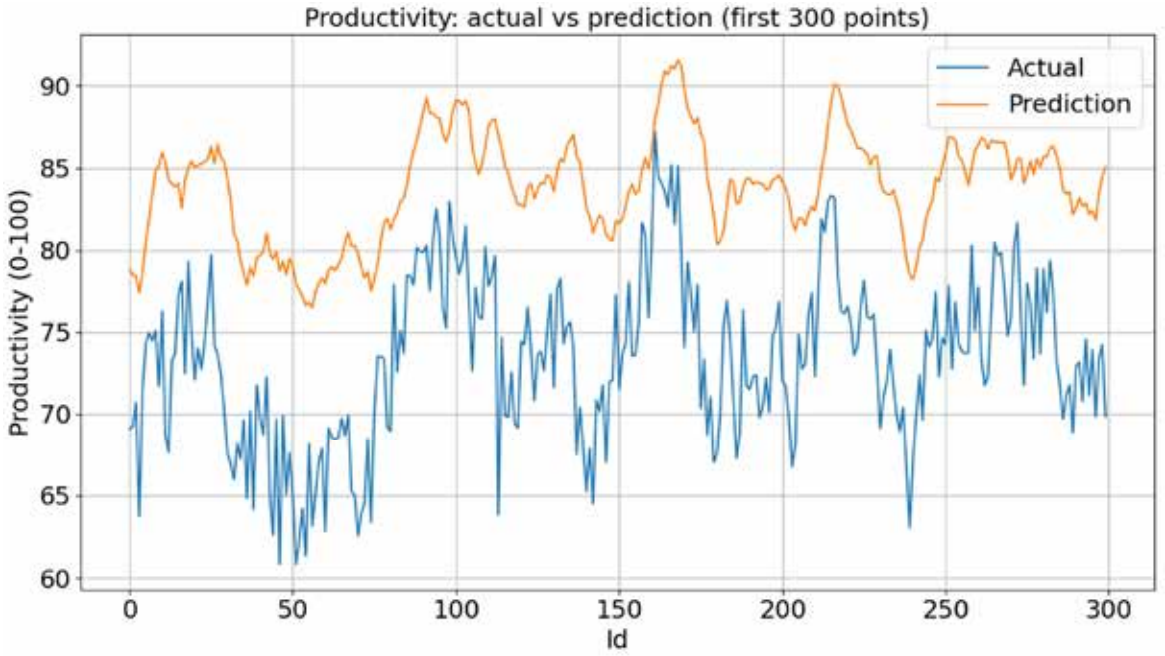

Figure 3. Forecast after data modification – repeating patterns appear

Source: compiled by the author based on own research and computational experiments

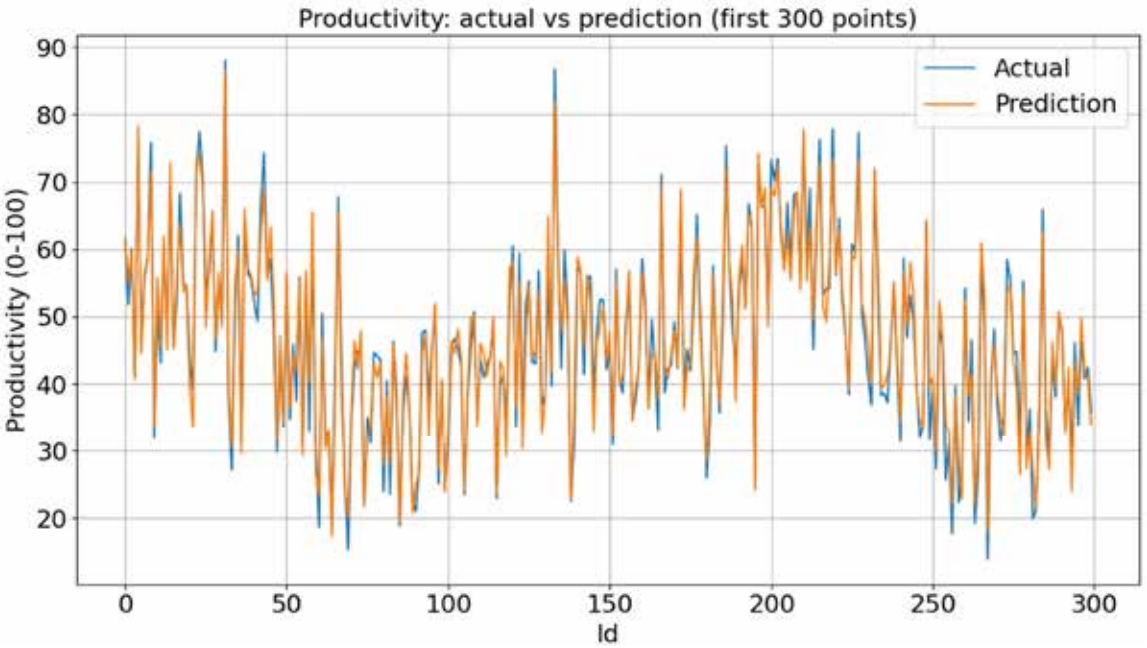

Figure 4. Final forecast – high correspondence to the fact

Source: compiled by the author based on own research and computational experiments

The results also indicate that LSTM models are able to capture long-term dependencies and nonlinear relationships in productivity time series, which makes them suitable for identifying trends, potential burnout phases, and workload fluctuations within project teams. This creates opportunities for proactive resource management and improved decision-making in HR analytics and project management systems.

At the same time, the research confirms that achieving stable forecasting performance requires sufficiently large and well-structured historical datasets of HR metrics. Therefore, the proposed approach is particularly relevant for medium and large organizations that systematically collect and analyze employee performance data.

The practical value of the study lies in the possibility of integrating the developed model with modern project management and team analytics tools (such as Jira APIs, Git activity logs, and performance review systems), enabling automated productivity forecasting and early detection of potential efficiency changes. Future research may focus on extending the model using hybrid neural architectures (e.g., LSTM combined with attention mechanisms), as well as validating the approach on real corporate datasets.

REFERENCES:

1. Sagheer A., Kotb M. Time series forecasting of petroleum production using deep LSTM recurrent networks. *Neurocomputing*. 2019. Vol. 323. P. 203–213. DOI: https://doi.org/10.1016/j.neucom.2018.09.082 (accessed November 12, 2025).

2. Nait Chabane A., Sahnoun M., Bettayeb B. Forecasting KPIs of production systems using LSTM networks. *Proceedings of the 9th International Conference on Data Analytics*. 2021. DOI: https://doi.org/10.1109/CyMaEn50288.2021.9497278 (accessed December 5, 2025).

3. Mahammad D. R., Priya S. L., Thulasimani T., Mann S., Mann S., Kalra G. Analysis and prediction of employee turnover characteristics based on LSTM-RNN-HMM model. *Journal of Intelligent & Fuzzy Systems*. 2025. DOI: https://doi.org/10.1201/9781003559115-56 (accessed December 18, 2025).

4. Giriprakash A., Arunadevi V., Rajalakshmi K., Khan S. A., Jeyalakshmi R., Valavan M. P. Achieving balanced allocation in human resources through LSTM framework method. *Proceedings of the 2024 IEEE International Conference on Smart Data Services*. 2025. DOI: https://doi.org/10.1109/ICISCN64258.2025.10934511 (accessed November 22, 2025).

5. Hochreiter S., Schmidhuber J. Long short-term memory. *Neural Computation*. 1997. Vol. 9, No. 8. P. 1735–1780. DOI: https://doi.org/10.1162/neco.1997.9.8.1735 (accessed November 3, 2025).

6. Greff K., Srivastava R. K., Koutník J., Steunebrink B. R., Schmidhuber J. LSTM: A search space odyssey. *IEEE Transactions on Neural Networks and Learning Systems*. 2017. Vol. 28, No. 10. P. 2222–2232. DOI: https://doi.org/10.1109/TNNLS.2016.2582924 (accessed December 14, 2025).

7. Goodfellow I., Bengio Y., Courville A. *Deep learning*. Cambridge: MIT Press, 2016. URL: https://www.deeplearningbook.org (accessed December 9, 2025).

8. Han J., Kamber M., Pei J. *Data mining: Concepts and techniques*. 3rd ed. Burlington: Morgan Kaufmann, 2011. DOI: https://doi.org/10.1016/C2009-0-61819-5 (accessed November 27, 2025).

9. Jolliffe I. T., Cadima J. Principal component analysis: A review and recent developments. *Philosophical Transactions of the Royal Society A*. 2016. Vol. 374, No. 2065. P. 20150202. DOI: https://doi.org/10.1098/rsta.2015.0202 (accessed November 16, 2025).

10. Breiman L. Random forests. *Machine Learning*. 2001. Vol. 45. P. 5–32. DOI: https://doi.org/10.1023/A:1010933404324 (accessed December 21, 2025).

11. Chollet F. *Deep learning with Python*. 2nd ed. Shelter Island: Manning Publications, 2021.

12. Ioffe S., Szegedy C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *Proceedings of the 32nd International Conference on Machine Learning*. 2015. P. 448–456. URL: https://arxiv.org/abs/1502.03167 (accessed November 13, 2025).

13. Géron A. *Hands-on machine learning with scikit-learn, Keras, and TensorFlow*. 3rd ed. Sebastopol: O'Reilly Media, 2022.

14. Werbos P. J. Backpropagation through time: What it does and how to do it. *Proceedings of the IEEE*. 1990. Vol. 78, No. 10. P. 1550–1560. DOI: https://doi.org/10.1109/5.58337 (accessed November 30, 2025).

15. Hyndman R. J., Athanasopoulos G. *Forecasting: Principles and practice*. 3rd ed. OTexts, 2021. URL: https://otexts.com/fpp3/ (accessed December 4, 2025).

16. Makridakis S., Wheelwright S. C., Hyndman R. J. *Forecasting: Methods and applications*. 3rd ed. New York: Wiley, 1998.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ:

1. Sagheer A., Kotb M. Time series forecasting of petroleum production using deep LSTM recurrent networks. *Neurocomputing*. 2019. Vol. 323. P. 203–213. DOI: https://doi.org/10.1016/j.neucom.2018.09.082 (дата звернення: 12.11.2025).

2. Nait Chabane A., Sahnoun M., Bettayeb B. Forecasting KPIs of production systems using LSTM networks. *Proceedings of the 9th International Conference on Data Analytics*. 2021. DOI: https://doi.org/10.1109/CyMaEn50288.2021.9497278 (дата звернення: 05.12.2025).

3. Mahammad D. R., Priya S. L., Thulasimani T., Mann S., Mann S., Kalra G. Analysis and prediction of employee turnover characteristics based on LSTM-RNN-HMM model. *Journal of Intelligent & Fuzzy Systems*. 2025. DOI: https://doi.org/10.1201/9781003559115-56 (дата звернення: 18.12.2025).

4. Giriprakash A., Arunadevi V., Rajalakshmi K., Khan S. A., Jeyalakshmi R., Valavan M. P. Achieving balanced allocation in human resources through LSTM framework method. *Proceedings of the 2024 IEEE International Conference on Smart Data Services*. 2025. DOI: https://doi.org/10.1109/ICISCN64258.2025.10934511 (дата звернення: 22.11.2025).

5. Hochreiter S., Schmidhuber J. Long short-term memory. *Neural Computation*. 1997. Vol. 9, No. 8. P. 1735–1780. DOI: https://doi.org/10.1162/neco.1997.9.8.1735 (дата звернення: 03.11.2025).

6. Greff K., Srivastava R. K., Koutník J., Steunebrink B. R., Schmidhuber J. LSTM: A search space odyssey. *IEEE Transactions on Neural Networks and Learning Systems*. 2017. Vol. 28, No. 10. P. 2222–2232. DOI: https://doi.org/10.1109/TNNLS.2016.2582924 (дата звернення: 14.12.2025).

7. Goodfellow I., Bengio Y., Courville A. *Deep learning*. Cambridge: MIT Press, 2016. URL: https://www.deeplearningbook.org (дата звернення: 09.12.2025).

8. Han J., Kamber M., Pei J. *Data mining: Concepts and techniques*. 3rd ed. Burlington: Morgan Kaufmann, 2011. DOI: https://doi.org/10.1016/C2009-0-61819-5 (дата звернення: 27.11.2025).

9. Jolliffe I. T., Cadima J. Principal component analysis: A review and recent developments. *Philosophical Transactions of the Royal Society A*. 2016. Vol. 374, No. 2065. P. 20150202. DOI: https://doi.org/10.1098/rsta.2015.0202 (дата звернення: 16.11.2025).

10. Breiman L. Random forests. *Machine Learning*. 2001. Vol. 45. P. 5–32. DOI: https://doi.org/10.1023/A:1010933404324 (дата звернення: 21.12.2025).

11. Chollet F. *Deep learning with Python*. 2nd ed. Shelter Island: Manning Publications, 2021.

12. Ioffe S., Szegedy C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *Proceedings of the 32nd International Conference on Machine Learning*. 2015. P. 448–456. URL: https://arxiv.org/abs/1502.03167 (дата звернення: 13.11.2025).

13. Géron A. *Hands-on machine learning with scikit-learn, Keras, and TensorFlow*. 3rd ed. Sebastopol: O'Reilly Media, 2022.

14. Werbos P. J. Backpropagation through time: What it does and how to do it. *Proceedings of the IEEE*. 1990. Vol. 78, No. 10. P. 1550–1560. DOI: https://doi.org/10.1109/5.58337 (дата звернення: 30.11.2025).

15. Hyndman R. J., Athanasopoulos G. *Forecasting: Principles and practice*. 3rd ed. OTexts, 2021. URL: https://otexts.com/fpp3/ (дата звернення: 04.12.2025).

16. Makridakis S., Wheelwright S. C., Hyndman R. J. *Forecasting: Methods and applications*. 3rd ed. New York: Wiley, 1998.

Дата надходження статті: 13.03.2026
Дата прийняття статті: 03.04.2026
Дата публікації статті: 07.04.2026