I.J. Intelligent Systems and Applications, **2025, 4, 50-57**
Published Online on August 8, 2025 by MECS Press (http://www.mecs-press.org/)
DOI: 10.5815/ijisa.2025.04.05

AI Scheduling with Contextual Transformers

Oleksandra Bulgakova*
P.M. Platonov Educational and Scientific Institute of Computer Engineering, Automation, Robotics, and Computer Programming, Odesa National University of Technology, Odesa, 65039, Ukraine
E-mail: sashabulgakova2@gmail.com
ORCID iD: https://orcid.org/0000-0002-6587-8573
*Corresponding author

Viacheslav Zosimov
P.M. Platonov Educational and Scientific Institute of Computer Engineering, Automation, Robotics, and Computer Programming, Odesa National University of Technology, Odesa, 65039, Ukraine
E-mail: zosimovvv@gmail.com
ORCID iD: https://orcid.org/0000-0003-0824-4168

Victor Perederyi
2Admiral Makarov National University of Shipbuilding, Mykolaiv, 54007, Ukraine
E-mail: viperkms1@gmail.com
ORCID iD: https://orcid.org/0000-0002-9241-3034

Received: 17 January 2025; Revised: 08 March 2025; Accepted: 28 April 2025; Published: 08 August 2025

Abstract: This paper presents a comprehensive framework for intelligent and personalized task scheduling based on Transformer architectures and contextual-behavioral feature modeling. The proposed system processes sequences of user activity enriched with temporal, spatial, and behavioral information to generate structured task representations. Each predicted task includes six key attributes: task type, execution time window, estimated duration, execution context, confidence score, and priority level. By leveraging Transformer encoders, the model effectively captures long-range temporal dependencies while enabling parallel processing, which significantly improves both scalability and responsiveness compared to recurrent approaches.
The system is designed to support real-time adaptation by integrating diverse data sources such as device activity, location, calendar status, and behavioral metrics. A modular architecture enables input encoding, multi-head self-attention, and global behavior summarization for downstream task generation. Experimental evaluation using artificially generated user data illustrates the model's ability to maintain high accuracy in task type and timing prediction, with consistent performance under varying contextual conditions. The proposed approach is applicable in domains such as digital productivity, cognitive workload balancing, and proactive time management, where adaptive and interpretable planning is essential.

Index Terms: Adaptive Scheduling, Transformer Model, Behavioral Sequence Modeling, Time Management, Productivity Optimization.

1. Introduction

In the digital environment, peoples are faced with an overwhelming number of tasks, notifications, and time-sensitive commitments, often leading to cognitive overload and inefficiencies in daily planning. Although various digital time management tools exist, most of them are static and fail to adapt to users' evolving contexts and behavior. These tools typically disregard crucial dynamic parameters such as location, productivity, shifting priorities, or mental and physical energy levels. As a result, schedules generated by such systems often require frequent manual adjustments and offer limited personalization.

Recent research has increasingly focused on intelligent time management solutions leveraging artificial intelligence. Notable efforts include systems that generate activity templates based on behavior-derived patterns in workplace settings [1], the incorporation of psychological and emotional factors into productivity analysis [2,3], and the use of AI in project management and automation [4-6,8]. Despite these advancements, many existing models still rely on basic algorithms or traditional recurrent neural networks (LSTM, GRU), which are often inadequate in capturing

This work is open access and licensed under the Creative Commons CC BY 4.0 License.

long-range temporal dependencies, integrating diverse data streams, or scaling efficiently for real-time applications.

This paper introduces an adaptive task scheduling framework that utilizes Transformer-based architectures to model user behavior in context-aware environments. Transformers, originally developed for natural language processing [7], are particularly suited for sequence modeling due to their multi-head attention mechanism, which captures global dependencies across long sequences in parallel. This makes them an effective solution for modeling complex behavioral and contextual patterns without the computational drawbacks of recurrent models.

A distinctive feature of our approach is its ability to personalize schedules dynamically based on real-time contextual data. The system processes multi-source inputs, including temporal markers, location, device status, behavioral metrics, and task history, to generate structured task recommendations. Each output includes six core attributes: task type, time window, estimated duration, execution context, confidence score, and priority level.

The goal is not only to improve the accuracy and adaptability of planning but also to enhance user experience by reducing manual effort and providing interpretable, context-sensitive task suggestions.

This paper is structured as follows: section 2 presents a review of the state of the art methods. Afterwards, section 3 presents a little description of the proposed Transformer-based scheduling methodology, including input encoding, model design, and task generation components. We analyze the simulation-based experiments results in section 4. In final, section 5 presents a conclusion of the paper.

2. Related Works

In this section, we review the state-of-the-art methods in intelligent task scheduling. These methods can be broadly divided into two categories: traditional approaches based on handcrafted rules or shallow machine learning algorithms, and deep learning-based approaches that extract temporal and contextual patterns from user behavior. We focus on recent developments in the second category, particularly on sequence modeling techniques such as RNNs, LSTMs, and Transformer-based architectures, and analyze their suitability for personalized scheduling in dynamic environments.

Intelligent task scheduling has evolved significantly, progressing from rule-based approaches to AI-driven adaptive systems. Early digital time management tools were grounded in expert-defined rules, static templates, and logical filters. These systems generated schedules based on fixed conditions such as the day of the week or task type [1,4,5], offering little adaptability and no mechanism for learning from user behavior over time.

To introduce automation, classical machine learning models such as decision trees, linear regression, and support vector machines were applied. While these approaches could handle a limited number of features, they lacked the capacity to detect long-term temporal dependencies or incorporate multi-source user context [13].

A notable improvement came with the adoption of recurrent neural networks and their variants such as LSTM and GRU [9-11,14]. These models made it possible to learn from sequences of user activity and predict upcoming tasks based on interaction history. However, RNN-based systems face inherent limitations: they struggle with long input sequences due to vanishing or exploding gradients, require strictly sequential data processing which reduces scalability, and are constrained in their ability to integrate diverse context sources (geolocation, device state, or energy levels).

To overcome these limitations, researchers have turned to Transformer architectures, originally developed for natural language processing tasks [12,17]. Transformers use a multi-head self-attention mechanism that captures dependencies between all elements in a sequence, regardless of their position, and allows for parallel processing making them highly suitable for time-sensitive behavioral modeling.

Advanced Transformer variants such as Informer [15] have shown strong performance in long-range forecasting, including time-series and scheduling tasks, while reducing computational complexity. These models are particularly effective in processing high-dimensional sequences with irregular or noisy input, a common challenge in real-world scheduling.

Several studies have proposed adaptive scheduling methods that combine user context, behavioral trends, and system feedback [2,3,16]. Such works emphasize the role of emotional states, cognitive load, and behavioral history in optimizing productivity. However, most existing models rely either on limited feature sets or non-adaptive architectures that do not support real-time personalization.

Furthermore, incorporating context awareness is essential for meaningful personalization. According to [13], context-aware systems should dynamically adjust behavior based on location, device state, availability, and user preferences. Nevertheless, many scheduling systems treat context as static filters rather than as dynamic, learnable components.

To address the identified limitations, we propose a scheduling architecture based on the Transformer model, designed for behavioral task planning. Our approach integrates sequential user behavior, temporal dynamics, spatial context, and task-specific metadata into a unified representation, enabling the generation of personalized and adaptive task schedules.

Our work addresses these gaps by proposing a Transformer-based architecture that integrates behavioral sequences, temporal and spatial context, and task metadata to generate personalized schedules. Unlike traditional or RNN-based methods, our system supports scalable, interpretable, and context-aware task planning adaptable to real-time user states.

3. Proposed Approach

3.1. Data Description

Let there be a historical sequence of user activity events over the past T steps (days or individual task executions), denoted as:

$$E = \{e_1, e_2, \dots, e_T\} \tag{1}$$

Each event e_T is a feature vector composed of four main groups of attributes $e_T = [A_t, B_t, C_t, D_t]$, where $A_t \in R^{d_A}, t = \overline{1..5}$ its *task-related features:*

A_1 – textual task description (title);
A_2 – category/type (work, rest, exercise…);
A_3 – task duration (in minutes);
A_4 – task importance (user-defined priority, scale 1–5);
A_5 – is completed (binary: 1 if completed, 0 otherwise).

$B_t \in R^{d_B}$ – *temporal features*:
B_1 – start time;
B_2 – day of week (0–6);
B_3 – time of day (morning, afternoon, evening, night);
B_4 – is weekend (binary);
B_5 – recurrence flag (binary: whether it is a recurring task).
$C_t \in R^{d_C}$ – *contextual features*:
C_1 – location (home, office, transit);
C_2 – device activity (screen on, idle, in motion);
C_3 – calendar availability (busy or free);
C_4 – interruptions count (number of interruptions during the task);
C_5 – weather code (sunny, rainy,…).
$D_t \in R^{d_D}$ – *behavioral metrics*:
D_1 – time avg (average focus duration in minutes)
D_2 – task completion rate (percentage of completed tasks);
D_3 – deviation from plan (average delay);
D_4 – energy profile (vector of productivity levels by hour);
D_5 – break frequency (average number of breaks per day).

The full input matrix is:

$$X = [e_1, e_2, \dots, e_T] \in R^{T \times d} \tag{2}$$

where $d = d_A + d_B + d_C + d_D$ is the total dimensionality of the embedded and encoded feature vector. The model aims to generate a sequence of predicted tasks for the next scheduling window $Z = \{\hat{z}_1, \hat{z}_2, \dots, \hat{z}_N\}$. Each predicted task $\hat{z}_i, i = \overline{1 \dots N}$ includes the following components: $\hat{z}_i^{(1)}$ – task type or title, $\hat{z}_i^{(2)}$ – predicted time window $[start_i]$, $\hat{z}_i^{(3)}$ – expected task duration, $\hat{z}_i^{(4)}$ – recommended execution context (e.g., location, device state), $\hat{z}_i^{(5)}$ – confidence score, $\hat{z}_i^{(6)}$ – optional priority score. The problem can be formulated as a multi-objective forecasting and sequence generation task:

$$f: R^{T \times d} \to R^{N \times m} \tag{3}$$

Where the function f is implemented by a transformer-based neural network model trained on sequential and contextual behavioral data. The model captures temporal dependencies and real-time context to output personalized schedule recommendations that adapt to individual user dynamics. The example of predicted output you can see in Table 1.

Table 1. The example of predicted output data

Task title, $\hat{z}_i^{(1)}$	**Time window**, $\hat{z}_i^{(2)}$	**Duration**, $\hat{z}_i^{(3)}$	**Context**, $\hat{z}_i^{(4)}$	**Confidence**, $\hat{z}_i^{(5)}$	**Priority**, $\hat{z}_i^{(6)}$
morning stand-up meeting	09:00 – 09:30	30 min	office	0.92	high
outdoor walk	16:00 – 16:45	45 min	outdoors	0.87	medium
respond to email messages	18:00 – 18:30	30 min	home	0.95	high

3.2. Methodology

To feed the user event sequence (2) into the Transformer encoder, each event vector e_T undergoes a fixed and clearly defined encoding procedure. The algorithm uses a consistent transformation pipeline for different feature types, chosen based on implementation requirements. The A_1 field is embedded using a pretrained fastText model (frozen during training). The resulting vector of dimension $d_{\text{text}} = 100$ represents the semantic content of the task:

$$\mathrm{A}_1^t = fastText(A_1^t) \epsilon R^{100} \tag{4}$$

All categorical variables are embedded using trainable embedding layers of fixed dimension $d_{\text{cat}} = 16$, including A_2, C_1, C_2, C_3, B_3. Each categorical value is converted into a dense vector via its respective embedding layer. Continuous numerical features, such as A_3, A_4, C_4, D_1, D_2, D_3, D_5 are standardized using fixed normalization parameters:

$$\tilde{x} = \frac{x-\mu}{\sigma} \tag{5}$$

where μ and σ are computed from the training set. Time-of-day values are encoded using sinusoidal time encoding:

$$B_1^t = [\sin(2\pi \cdot t/24), \cos(2\pi \cdot t/24)] \tag{6}$$

D_4 given as a 24-dimensional vector (one per hour), is projected with a linear layer to a fixed-size vector of 32 dimensions:

$$D_4^t = W_{proj} D_4 + b \tag{7}$$

All encoded components from steps 1 to 5 are concatenated into a unified vector of fixed length d=256

$$\tilde{e}_t = Concat(A_1^t, A_2^t, \dots, D_5^t) \epsilon R^{T \times 256} \tag{8}$$

The complete input sequence becomes:

$$X = [\tilde{e}_1, \tilde{e}_2, \dots, \tilde{e}_T] \epsilon R^{T \times 256} \tag{9}$$

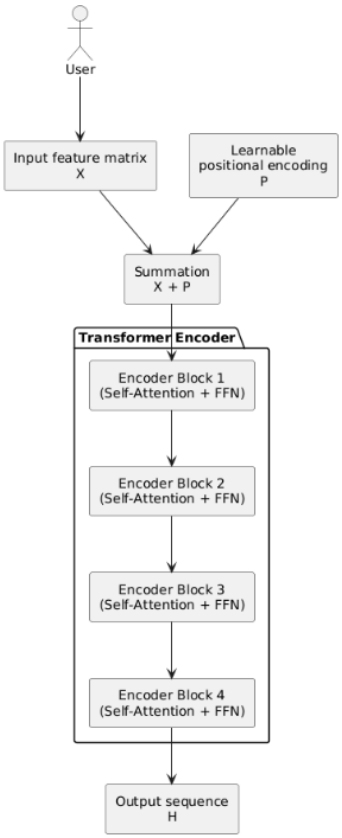


Fig.1. Transformer encoder architecture

The Fig.1 illustrates the internal structure of the Transformer encoder used in the proposed adaptive planning system. The model receives an input sequence of feature vectors X, representing the user's historical events. First, learnable positional encodings P are added to the input vectors to preserve the order of events within the sequence. The result $X + P$ is then passed to a stack of four identical encoder blocks.

Each encoder block consists of a multi-head self-attention mechanism that computes attention weights between all elements in the sequence; a feed-forward neural network that transforms the representations; and residual connections with layer normalization, which improve learning stability.

The encoder produces an output matrix H, where each vector h_t contains a contextual representation of event e_t, enriched with global sequence information.

The resulting representations $H = [h_1, h_2, \dots, h_T]$ are high-level contextual vectors that encode the user's behavioral history in a format suitable for predicting future actions. Each vector h_t contains not only the information about the corresponding input event e_t, but also the influence of all other events in the sequence both recent and distant. Thanks to the self-attention mechanism, the model can determine which past events are most relevant for forecasting.

These contextual embeddings are then passed to the next stage – the Task Generation Module, where the encoder output is transformed into explicit recommendations: task types, optimal time windows, durations, expected execution contexts, and priorities.

The task generation module is responsible for converting the output representations H, produced by the Transformer encoder, into a personalized set of predicted tasks $Z = \{\hat{z}_1, \hat{z}_2, \dots, \hat{z}_N\}$ for the upcoming scheduling interval. Each task is defined by six parameters, N=6. The module first aggregates the encoder outputs using a global attention mechanism. Let W_q, W_k, and W_v denote the trainable projection matrices for the query, key, and value representations, respectively. Then the global vector is computed as:

$$\alpha = softmax\left(\frac{(HW_q)(HW_k)^T}{\sqrt{d}}\right), h_{global} \tag{10}$$

The resulting global vector h_{global} ummarizes the user's recent behavior and serves as input for task generation. For each task i, a prediction vector $\hat{z}_i$ is produced as follows. The task type is predicted with a softmax classifier:

$$\hat{z}_i^{(1)} = argmax\ \left(softmax\left(W_1 h_{global} + b_1\right)\right) \tag{11}$$

The task start and end times are generated using a regression head:

$$\hat{z}_i^{(2)} = \left[\hat{t}_{start}^i, \hat{t}_{end}^i\right] = W_2 h_{global} + b_2 \tag{12}$$

The duration can be either predicted independently:

$$\hat{z}_i^{(3)} = W_3 h_{global} + b_3 \tag{13}$$

or derived as the difference $\hat{t}_{end}^i - \hat{t}_{start}^i$, depending on the implementation. The expected execution context is also predicted using a *softmax* classifier:

$$\hat{z}_i^{(4)} = argmax\ \left(softmax\left(W_4 h_{global} + b_4\right)\right) \tag{14}$$

The model estimates the likelihood of task relevance through a sigmoid activation:

$$\hat{z}_i^{(5)} = \sigma\left(W_5 h_{global} + b_5\right) \tag{15}$$

The predicted priority is generated via an additional *softmax* output:

$$\hat{z}_i^{(6)} = argmax\ \left(softmax\left(W_6 h_{global} + b_6\right)\right) \tag{16}$$

Each task $\hat{z}_i$ is generated independently but is based on the shared global representation h_{global}, which ensures overall coherence and personalization. The generation module transforms abstract behavioral embeddings into explicit, interpretable scheduling elements, enabling a personalized and explainable planning experience.

The proposed intelligent planning system is trained on historical user data, which includes sequences of completed tasks, temporal markers, execution contexts, and behavioral metrics. Training is conducted in a supervised learning setting with a multi-output architecture that combines regression and classification objectives.

The model is optimized using a composite loss function that integrates several components. Classification targets such as task type, execution context, and priority are optimized with cross-entropy loss. Regression targets such as the

time window and task duration are optimized using mean squared error. The confidence score, reflecting the predicted relevance of a task, is trained via binary logistic loss. The total loss function is a weighted sum of all components. Following initial training on large-scale historical datasets, the system supports personalization through feedback-based adaptation. All user interactions including acceptance, rejection, rescheduling, or editing of tasks are logged and can be utilized for model fine-tuning either on-device or within a secure environment.

Adaptation may be performed in batch mode (e.g., periodic local weight updates) or in an online setting, for example by fine-tuning the final layers or applying gradient-based refinement of predictions. This allows the model to retain its generalization capability while improving responsiveness to individual patterns. The architecture is also compatible with active learning techniques, whereby the system can request user input to refine preference modeling.

4. Experiments

To preliminarily validate the functionality of the proposed algorithm, we conducted a series of experiments using artificially generated user data simulating hypothetical users with varying patterns of activity. This approach allowed us to test the model architecture in a fully controlled setting, verify its robustness to noise, and confirm its ability to learn behavioral dependencies even without prior exposure to real-world datasets.

4.1. Databases and Evaluation Metrics

The artificial data consisted of task sequences of length $T = 100$, with each task comprising structured features such as task type, start time, duration, importance, execution context, and behavioral attributes. The Table 2 presents a fragment of input data.

Table 2. The fragment of input data

A_1	A_2	A_3	A_4	A_5	B_1	…	C_1	…	D_1	..	D_3	D_4
email check	work	30	4	1	08:30	…	home	…	25	…	5	0.9
workout	sport	45	3	1	10:00	…	gym	…	40	…	0	0.8
team sync	meetig	60	5	1	11:30	…	office	…	50	…	10	0.7
…	…	…	…	…	…	…	…	…	…	…	…	…
lunch	rest	60	2	1	13:00	…	cafe	…	20	…	0	0.6
code review	work	90	5	0	14:30	…	office	…	55	…	15	0.85
…	…	…	…	…	…	…	…	…	…	…	…	…

4.2. Results

The model generated a personalized schedule for the upcoming time window. Since the structure of the input data and reference outputs were predefined, predictions could be directly compared to the ground truth based on task type, start time, duration, and execution context in Table 3.

Table 3. The fragment of output data

$\hat{z}_i^{(1)}$	$\hat{z}_i^{(2)}$	$\hat{z}_i^{(3)}$	$\hat{z}_i^{(4)}$	$\hat{z}_i^{(5)}$	$\hat{z}_i^{(6)}$
work	08:30 – 09:00	30	home	0.92	medium
sport	10:00 – 10:45	45	gym	0.89	medium
meetig	11:30 – 12:30	60	office	0.93	high
…	…	…	…	…	…
rest	13:00 – 14:00	60	cafe	0.88	low
work	14:30 – 16:00	90	office	0.85	high

To better understand how the model performs in practice, we analyzed where it tends to make mistakes, Table4. Overall, the system predicted task types, timing, and context very accurately, almost all scheduled tasks matched the input data in these aspects. However, we noticed that the model sometimes made different decisions about task priority. In about 15% of the cases, it marked important tasks as only “medium” priority. This could be due to how the model interprets the broader situation, such as task congestion or time availability.

We also looked at how well the model predicts how long tasks will take and when they should start. On average, the timing was very close to the expected values, with only minor differences, just a few minutes off, which is acceptable in most real-life planning scenarios.

We checked how confident the model was in its predictions. Most tasks had high confidence scores, showing that the model was generally sure about its suggestions.

These results show that the system works well in structured environments with clear behavior patterns. The few mistakes it makes seem reasonable and are likely related to the limitations of synthetic data. As a next step, we plan to

test the model on real-world user data to evaluate its adaptability to natural behavior patterns, unexpected task types, and incomplete or noisy input. This will allow us to assess the model's performance in realistic conditions and improve its personalization mechanism

Table 4. Quantitative error analysis on simulated data

Metric	Value
Task type prediction accuracy	98%
Priority prediction accuracy	85% 15% tasks downgraded to "medium" priority
MAE (task duration)	3.1 min
MAE (time window start/end)	4.4 min
Mean confidence score	0.894
Standard deviation	0.03

5. Conclusions

This paper introduced an intelligent task planning architecture based on a Transformer encoder and contextual-behavioral feature modeling. The proposed system is designed to generate personalized task recommendations by encoding user activity sequences and dynamically predicting structured task vectors that include type, time window, duration, context, confidence, and priority.

To validate the model's functionality, we conducted a controlled experiment on artificially generated data. This allowed us to evaluate the behavior of the model in a noise-free environment and test its ability to detect patterns in synthetic user behavior. The model achieved high accuracy in task type prediction and timing estimation, correctly aligning most predicted outputs with the corresponding input events. Some variation was observed in priority predictions, which may reflect the model's consideration of additional contextual signals beyond raw importance scores.

The findings demonstrate that the model architecture is capable of producing interpretable and contextually appropriate task suggestions based on behavioral sequences. However, the results also highlight that performance on artificial data cannot be fully generalized to real-world scenarios. In real settings, the presence of behavioral noise, unpredictability, and personal variability necessitates continuous adaptation and learning from feedback.

In future work, we plan to deploy the proposed model in real-world environments by collecting behavioral data from users through mobile applications and wearable devices. This will enable evaluation of the system's adaptability to natural, unstructured behavior patterns. To address challenges related to incomplete or noisy data, we will implement imputation strategies and assess model stability under uncertainty.

References

[1] Kumar P., Kumar R., Nagesh B, et al .Advancing Digital Payment Systems: Combining AI, Big Data, and Biometric Authentication for Enhanced Security[J]. Engineering and Computer Science, 2022, 11(8). doi: 10.18535/ijecs/v11i08.4698.

[2] Mario Alvarez-Jimenez, Peter Koval, Lianne Schmaal，et al. The Horyzons project: a randomized controlled trial of a novel online social therapy to maintain treatment effects from specialist first-episode psychosis services[J]. World Psychiatry, 2021, 20(2). doi: 10.1002/wps.20858.

[3] Chelsea Arnold, John Farhall, Kristi-Ann Villagonzalo et al. Engagement with online psychosocial interventions for psychosis: A review and synthesis of relevant factors[J]. Internet Interventions, 2021, 25:100411. doi: 10.1016/j.invent.2021.100411

[4] André Barcaui, André Monat. Who is better in project planning?Generative artificial intelligence or project managers?[J]. Project Leadership and Society, 2023, 4(100101). doi: 10.1016/j.plas.2023.100101

[5] Muhammad Irfan Hashfi; Raharjo, Teguh. Exploring the Challenges and Impacts of Artificial Intelligence Implementation in Project Management: A Systematic Literature Review[J]. Advanced Computer Science and Applications, 2023, 14(9): 366-376.

[6] Zosimov V., Bulgakova O. Calculation the Measure of Expert Opinions Consistency Based on Social Profile Using Inductive Algorithms. 15th International Scientific Conference on Intellectual Systems of Decision Making and Problems of Computational Intelligence, ISDMCI 2019, 2019, 1020: 622-636. doi: 10.1007/978-3-030-26474-1_43.

[7] C. Subakan, M. Ravanelli, S. Cornell, M. Bronzi and J. Zhong. Attention Is All You Need In Speech Separation. ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Toronto, ON, Canada, 2021, pp. 21-25, doi: 10.1109/ICASSP39728.2021.9413901.

[8] V. Abhishek, S. Binny, T. Johan , et al. Federated Learning: Collaborative Machine Learning without Centralized Training Data[J]. Engineering Technology and Management Sciences, 2022, 5(6): 355-359.

[9] Das, S., Tariq, A., Santos, T., Kantareddy, S.S., Banerjee, I. (2023). Recurrent Neural Networks (RNNs): Architectures, Training Tricks, and Introduction to Influential Research. In: Colliot, O. (eds) Machine Learning for Brain Disorders. Neuromethods, vol 197. Humana, New York, NY. https://doi.org/10.1007/978-1-0716-3195-9_4.

[10] V. Zosimov and O. Bulgakova. Application of Personalized Ranking Models Based on Expert Evaluations for Sorting Goods on E-commerce Web Resources. 2020 IEEE 15th International Conference on Computer Sciences and Information Technologies (CSIT), Zbarazh, Ukraine, 2020, pp. 42-45, doi: 10.1109/CSIT49958.2020.9321902.

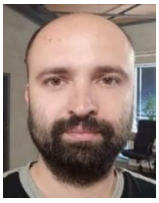
[11] Nosouhian, S., Nosouhian, F., & Kazemi Khoshouei, A. A Review of Recurrent Neural Network Architecture for Sequence Learning: Comparison between LSTM and GRU. Preprints. 2021. doi: 10.20944/preprints202107.0252.v1.

[12] Niki Parmar, Ashish Vaswani, Jakob Uszkoreit, Lukasz Kaiser, Noam Shazeer, Alexander Ku, Dustin Tran Proceedings of the 35th International Conference on Machine Learning, PMLR 80:4055-4064, 2018.
[13] G.D. Abowd, A.K. Dey, P.J. Brown, N. Davies, M. Smith, P. Steggles. Towards a Better Understanding of Context and Context-Awareness. In: Gellersen, HW. (eds) Handheld and Ubiquitous Computing. HUC 1999. Lecture Notes in Computer Science, vol 1707, 1999. Springer, Berlin, Heidelberg. doi: 10.1007/3-540-48157-5_29
[14] S. Hochreiter, J. Schmidhuber. Long Short-Term Memory. Neural Computation, 1997, 9(8), 1735–1780. doi: 10.1162/neco.1997.9.8.1735
[15] H. Zhou, S. Zhang, J. Peng, et al. Informer: Beyond Efficient Transformer for Long Sequence Time-Series Forecasting. Proceedings of the AAAI Conference on Artificial Intelligence, 2021, 35(12). doi: 10.1609/aaai.v35i12.17325
[16] G. Taye, S. Sharma, P. Shah and Y. G. Nuriye. Exploring the Role of Artificial Intelligence in Class Scheduling and Management: A Comprehensive Survey and Review. 2023 International Conference on Computer Science and Emerging Technologies (CSET), Bangalore, India, 2023, pp. 1-11, doi: 10.1109/CSET58993.2023.10346898
[17] A. Vaswani, N. Shazeer, N. Parmar, N. et al. Attention is All You Need. Advances in Neural Information Processing Systems, 2017, 30. doi: 10.48550/arXiv.1706.03762

Authors' Profiles



Dr. Oleksandra Bulgakova is a researcher at the P.M. Platonov Educational and Scientific Institute of Computer Engineering, Automation, Robotics, and Computer Programming of Odesa National University of Technology, Odesa, Ukraine. She holds a PhD Degree in Computer Science (International Research and Training Center for Information Technologies and Systems of the NAS and MES of Ukraine in 2011). Her research interests include inductive modeling, particularly GMDH and hybrid algorithms, as well as web mining and web-data analysis. She focuses on applying neural networks and generalized inductive algorithms (GIA) in e-commerce and e-learning to enhance personalization and intelligent decision-making in digital systems.



Dr. Viacheslav Zosimov is a researcher at the P.M. Platonov Educational and Scientific Institute of Computer Engineering, Automation, Robotics, and Computer Programming of Odesa National University of Technology, Odesa, Ukraine. He holds a DSc Degree in Computer Science (Taras Shevchenko National University of Kyiv in 2020). His research interests include ontologies and semantic web technologies, with a focus on structuring knowledge for intelligent web systems. He also explores object-oriented programming in the development of adaptive applications for e-commerce. Additionally, he is interested in inductive modeling methods for data-driven decision support and system optimization.



Dr. Victor Perederyi received his DSc Degree in models, methods, and information technologies for decision-making in organizational and technical systems from Kherson National Technical University, Ukraine, in 2021. He has been a professor in the National University of Shipbuilding since 2022. His research interests focus on the development of models, methods, and information technologies for decision-making in organizational and technical systems, particularly those designed for critical applications. This includes intelligent decision support, real-time data processing, risk assessment, and system optimization under uncertainty. He is also interested in the integration of inductive modeling, simulation tools, and adaptive algorithms to improve the reliability and performance of mission-critical infrastructures.

How to cite this paper: Oleksandra Bulgakova, Viacheslav Zosimov, Victor Perederyi, "AI Scheduling with Contextual Transformers", International Journal of Intelligent Systems and Applications(IJISA), Vol.17, No.4, pp.50-57, 2025. DOI:10.5815/ijisa.2025.04.05