VOLUME 4 NUMBER 2 2026 ISSN 2958-0846
eISSN 2958-0854

AL-FARABI KAZAKH NATIONAL UNIVERSITY

Journal of Problems in Computer Science and Information Technologies

№2 (4) 2026

Al-Farabi Kazakh National University

ISSN: 2958-0846; eISSN: 2958-0854

Journal of Problems in Computer Science and Information Technologies №2 (4) 2026

03.02.2023 Registered with the Ministry of Information and Social Development of the Republic of Kazakhstan

№ KZ61VPY00064018

The journal is published 4 times a year
(March, June, September, December).

IB № 18097. Signed to publishing 25.06.2026. Format 60x84/8. Offset paper.
Digital printing. Volume 8,6 printer's sheet. Edition: 300. Order №3580.
Publishing house «Kazakh University»
www.read.kz Telephone: +7 (727) 3773330, fax: +7 (727) 3773344
Al-Farabi Kazakh National University KazNU, 71 Al-Farabi, 050040, Almaty
Printed in the printing office of the Publishing house «Kazakh University».

IRSTI 28.23.25 https://doi.org/10.26577/jpcsit4220261

Dinara Zhussupova

Institute of Mathematics and Mathematical Modeling, Astana, Kazakhstan
e-mail: zhus.dinara@gmail.com

APPLYING SELF-PLAY REINFORCEMENT LEARNING TO KAZAKH NATIONAL BOARD GAMES: A CASE STUDY ON TOGYZKUMALAK

Abstract. The paper presents the results of a study on training agents for competitive board games, using the Kazakh national Togyzkumalak game, a zero–sum game, as a case study. As demonstrated by Alpha Zero, it is possible to train a champion without human expert knowledge or supervised learning. However, each game has unique characteristics that require specific research to develop an excellent player. In this study, two approaches were used to train agents: model-free and model-based. The self-play technique was used effectively during the training. As a result, the agents developed the ability to play Togyzkumalak at a competitive level.

Keywords: Multi-agent reinforcement learning, cumulative reward, Q-learning, zero-sum game, Monte Carlo tree search.

1. Introduction

Multi-agent learning uses reinforcement learning algorithms, but it is a significantly more complex area of agent learning. In reinforcement learning (RL), the agent's goal during interaction with the environment is to maximize the reward assigned to it at each moment in time [1]. The reward signal helps the agent determine whether it is acting correctly in the environment or if it needs to adjust its policy moving forward.

In multi-agent reinforcement learning (MARL), the problem is complicated by the fact that multiple agents in the environment may have different goals and varying knowledge about the environment. In addition, each agent's reward may depend on other players' actions and policies.

The concept of cumulative reward helps to evaluate the future rewards of actions taken at the current time. This type of reward informs the agent about how promising a particular action is in the short term and how optimal it is over the long term. The concepts of cumulative reward and value function are central to most reinforcement learning methods.

Reinforcement learning for multi-agent environments [2] is based on RL, where agents learn the optimal policy through trial and error, aiming to maximize the received reward. While single-agent RL focuses on learning the optimal policy for an individual agent, MARL addresses learning optimal policies for multiple agents and the unique challenges that arise in this process.

An important feature of MARL is the non-stationarity caused by the constantly changing policies of agents during learning. This non-stationarity can lead to a moving target problem, as each agent adapts to the policies of others, whose policies are also continuously adjusting in response. This can result in cyclical and unstable learning dynamics.

The ability to robustly bandle such non-stationarity is thus often a critical aspect of MARL algorithms and has been the focus of extensive research [3], [4], [5].

In RL, an agent's policy is considered optimal if it yields the highest reward in each state. In a multi-agent environment, however, the optimal policy for one agent depends on the policies of other agents due to their competitiveness or cooperativeness. Therefore, a more comprehensive approach is necessary. In Chapter 4 of the book [2] there are various approaches to setting the problem and solving them for several agents interacting in one environment. Additionally, unlike in RL, where the optimal policy consistently provides the maximum reward, in MARL the situation is more complex, since an agent's reward also depends on the actions of other agents. The main goal of MARL is to develop learning methods that lead to policies that merge to a solution.

MARL is essentially RL applied to multi-agent game models to learn optimal policies for agents. Thus, MARL is based on both RL theory and game theory. Agents interact within an environment to achieve certain goals. This concept can be defined

using models of multi-agent interaction. These models, based on game theory, are commonly referred to as games.

Minimax is a solution concept used in two-agent zero-sum games where the reward of one agent is negative compared to the reward of the other. The existence of a minimax solution for normal-form games is discussed in Neumann [6], [7]. The minimax problem can be solved using linear programming algorithms, such as the simplex method [8].

Independent learning often uses the self-play technique [9], [10], [11]. Although independent learning usually implicitly assumes self-play (e.g., in independent Q-learning), it is not strictly required as agents can use different algorithms within this approach. Self-play has been used effectively to train Alpha Zero AI [12] and to develop competitive agents capable of learning complex motor skills such as wrestling and ball-playing [13]. Recent work has introduced several multi-agent tasks with competing goals in a 3D world with simulated physics in the MuJoCo framework [14], where agents must learn complex motor skills to thrive in a competitive environment.

A recent breakthrough in MARL was the victory of the OpenAI team-trained agents in the game DOTA2 [15] against a team of professional players. However, it is important to note that this victory was not entirely unrestricted, as there were limitations on role selection, locations, and some features for players.

The next advancement in this field was to train agents without any prior knowledge of the game.

MuZero AI was presented by Google DeepMind researchers in the work [16].

This paper explores methods for training agents to play Togyzkumalak. Structure of the paper:

- Section 2 contains information about the game and the rules;
- Section 3 outlines the challenges of training an agent for Togyzkumalak, including search complexity, game depth, chaotic mechanics, and the need for extensive self-play;
- Research methods are described in Section 4;
- Section 5 describes the development tools used in the research;
- Section 6 presents the results for two algorithms;
- Future research and interesting conclusions from the obtained results are described in Section 7.

2. About game

Togyzkumalak is one of the traditional Kazakh national games. It is a strategic board game played on a special board with 162 stones. In 2020, UNESCO included Togyzkumalak in the Representative List of Intangible Heritage.

2.1 Description

The game of Togyzkumalak is played between two players on a special board. The board features two accumulation holes and 18 game holes (see Figs. 1 and 2). During the game, each player controls one accumulation hole. At the start of the game, 162 stones are distributed across the 18 game holes with 9 stones in each hole.

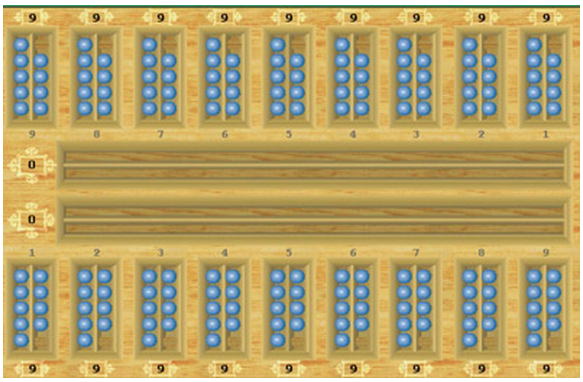
The player who makes the first move is called ”Bastaushy (Beginner)”, while the player who makes the subsequent move is called ”Kostaushy (Continuer)”. Sometimes, the terms ”white side” and ”black side” are used to refer to the Beginner and Continuer, respectively.

The general structure of the board is shown in Figs. 1 and 2.

In Fig. 2, the player’s game holes are positioned closer to them, while the accumulation hole, called ”Kazan”, is located opposite the opponent. The number of stones in the Kazan is indicated at the bottom of the cell, and the total number of stones it contains there is also displayed.

Figure 1
Wooden board with ornaments

Figure 2
The standard game board



2.2 Basic rules of the game

The game is played by two competing players. Togyzkumalak is a zero-sum game, i.e. the stones won by one player are effectively subtracted from the other player's total.

Before the game starts, the board is set up in the starting position. Here are the basic rules of the game.

1) Description of a player's move.

- The player takes all the stones except one from one hole and places one stone in each cell in a counterclockwise direction. The move must be made only from a non-empty hole; if all the holes are empty, this position is called "Atsyrau" (see 3).

- If the last hole contains an even number of stones and is on the opponent's side, the player captures these stones and places them in their accumulation hole.

Exception. If there is only one stone in the current hole, it is moved to the adjacent hole on the right.

Exception. You cannot move stones from a hole captured by your opponent, referred to as from Tuzdyk.

2) Getting a Tuzdyk. Tuzdyk is a hole captured from your opponent. Let us look at cases where a player gets a Tuzdyk:

a) if the last stone placed does not land in the opponent's final cell and the number of stones in that hole becomes 3, the player captures as a Tuzdyk

b) it is not allowed to capture a Tuzdyk in the same numbered hole as the opponent's Tuzdyk (for example, if your opponent has captured hole number 3, you cannot capture a hole with the same number).

3) Atsyrau. A situation in the game when a player cannot make any move because there are no stones left in their game holes.

4) Winning the game.

First case: A player wins if they collect more than 81 stones in their accumulation hole.

Second case: A player also wins if their opponent reaches Atsyrau, in which case all remaining stones in the opponent's game holes are transferred to the player's accumulation hole.

3. Problem formulation

The difficulty of training an agent for the Togyzkumalak game can be described in terms of combinatorial complexity, game depth, and state-space dimensionality.

3.1 Game tree size

At the beginning of the game, a large number of possible moves significantly increase the size of the search tree.

In chess, the average branching factor is ≈ 35. In Togyzkumalak:

- at the start, there are 9 possible moves (choosing from one of the 9 holes);

- after a few moves, Tuzdyks appear, introducing new rules;

- in the middle of the game, there are typically 5–7 meaningful moves.

Although the number of possible branches in the initial position may be lower than in chess, it does not narrow significantly due to the complex rules of stone redistribution.

3.2 Depth of play – length of matches

Games in Togyzkumalak can last more than 100 moves, which is significantly longer than in chess, where the average game length is around 40–50 moves. In Togyzkumalak, the games often exceed 100 moves.

The average game length in Go is about 200-250 moves, but can vary greatly. If both players fight for every territory until the end, a game can extend beyond 300 moves.

Although Go games are much longer than those in Togyzkumalak, the branching factor (number of possible moves) is lower due to the game's territorial nature.

Deep games make deep learning more challenging because mistakes made early on may only become apparent 50–60 moves later.

3.3 Determinism vs. stochasticity

Togyzkumalak is a deterministic game, but the redistribution of stones after a move creates complex effects that are difficult to predict.

In chess, a player simply moves a piece. In Togyzkumalak, a single move can immediately redistribute 10+ stones.

This characteristic introduces a level of chaos, making position evaluation more challenging for a neural network. The same action can lead to completely different results depending on the distribution of stones on the board.

3.4 Position evaluation

In chess, piece differences can be used as a simple heuristic.

At the beginning of a Togyzkumalak game, material balance has little impact since both players have the same number of stones.

Key moments in the game revolve around Tuzdyk and Kazan management, which are harder to encode as a simple function. A trained neural network must understand strategic concepts such as:

- control over Tuzdyk
- advantageous stone redistribution
- defending and attacking the opponent's game hole.

Therefore, deep neural network architectures are needed to evaluate positions effectively.

3.5 Self-play training problem

At the beginning of training, the agent plays poorly and many games end randomly.

Compared to chess, where more natural patterns emerge (such as center control and piece development), mistakes in Togyzkumalak can appear random (e.g., losing a Tuzdyk early in the game).

As a result, significantly more self-play games are required before the agent begins to learn meaningful strategies.

4. Methods

While independently training agents for the Togyzkumalak game, two RL approaches can be used: model- free and model-based algorithms. The neural network architectures were selected on the basis of the experiment results. Since the representation of the board – the state of the game – differs between the methods, the descriptions are provided separately in Subsections 4.1 and 4.2.

4.1 Deep Q-network

The game board represents data on the number of stones in each game hole and the accumulation holes, as well as the number of Tuzdyks. Each game state is therefore described by a 23-dimensional vector, where the first 18 components represent the number of stones in the game holes, the 19th and 20th components represent the number of stones collected in the accumulation holes by the first and second players, respectively, and the 21st and 22nd components represent the numbers of holes won as Tuzdyks by each player. If no Tuzdyk has been captured, the value is set to -1. The last coordinate of the vector indicates the player's number.

In the game, the state space is discrete, a tensor of dimension 23 × 1, and the action space is also discrete, a tensor of dimension 9 × 1. According to the rules, a player cannot move from a hole that has no stones. Therefore, when implementing the game environment, action masking is used.

A game with a person requires displaying the state after each move. For this purpose, it is convenient to represent the state in graphical form Figure 3.

Figure 3
Representation of the board in the starting position

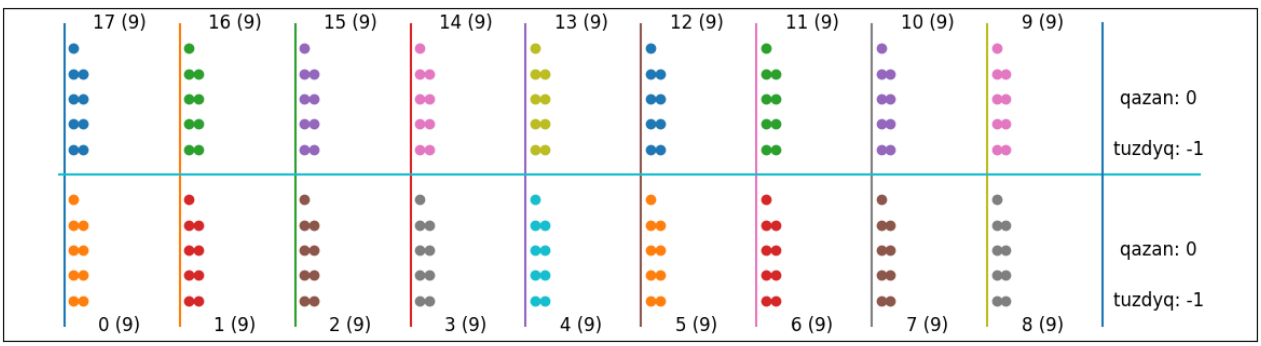
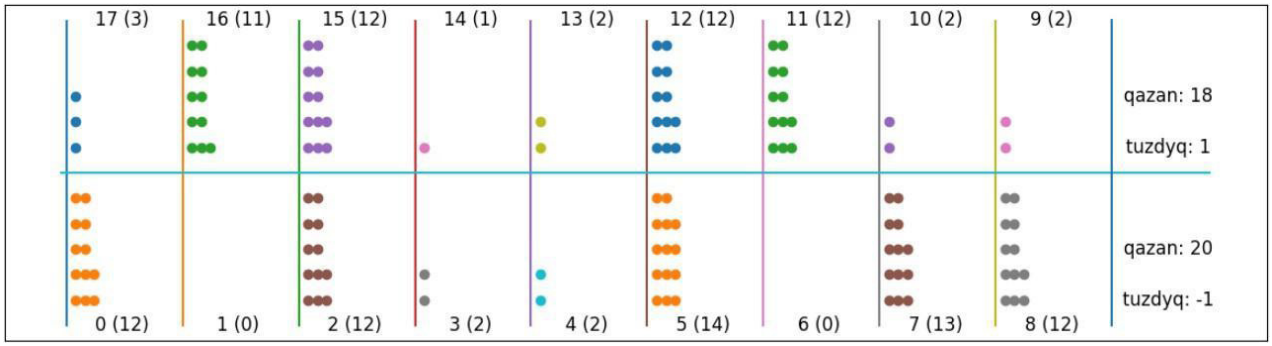


Figure 4
Representation of the board during the game



The environment for the Togyzkumalak game was implemented using the PettingZoo framework [17]. Figures 3 and 4 show graphical representations of the initial and playing positions of the game. The number of stones in the accumulation holes and the number of Tuzdyk cells are indicated on the right.

The Deep Q-Learning algorithm, also known as Deep Q-network [18] was used to train the agents by estimating the Q-value, which represents the expected reward. Thus, the Q-value corresponds to the expected reward for performing the selected action in the current state, following a particular policy. Deep Q-Learning uses a deep neural network to approximate the Q-values for each possible action-state pair. To optimize the learning process, a modified version, Double Deep Q-Learning [19] was used instead of the standard DQN.

For the Double DQN algorithm, it is very important to choose the most optimal neural network architecture. In the self-play technique described in Section 1, the weights are loaded from the previous model, so the choice must be made at the beginning.

The following MLP architectures were considered: NET ARCHS = [[128, 256, 256, 128], [256, 512, 256], [256, 512, 512, 256], [512, 1024, 512], [1024, 2048, 2048, 1024], [2048, 4096, 4096, 2048], [2048, 4096, 8192, 4096, 2048]].

ReLU activation function is used between the linear layers. The exploration and subsequent selection were carried out using neural network training with random policies as adversaries.

The results and details of the training are given in Subsection 6.1.

4.2 Monte Carlo Tree Search

The Monte Carlo Tree Search (MCTS) algorithm was originally developed to train agents to play Go [12]. The AI, called AlphaZero Go, used a deep neural network. The algorithm was further developed in [20], and its application was extended to other games, such as Chess and Shogi. The neural network used in the algorithm allowed for a reduction in the number of roll-outs during gameplay by two orders of magnitude. Specifically, AlphaZero searches only 80,000 positions per second in Chess and 40,000 in Shogi, compared to 70 million for Stockfish [21] and 35 million for Elmo. AlphaZero compensates for the smaller number of evaluations by using its deep neural network to focus much more selectively on the best option. Since AlphaZero AI defeated the previous champions in each of the games it completed in (AlphaZero Go in Go, Stockfish and Elmo in Chess and Shogi respectively), it is currently considered the world champion.

AlphaZero was trained solely through self-play.

The MCTS algorithm in this study was applied to the Togyzkumalak game. The game board is implemented as in Figure 5. The first line indicates the number of stones in the players' accumulation holes, while lines 2 through 10 show the number of stones in the game holes. The left side belongs to the first player and the right side to the second player.

In the board representation, "o" denotes stones belonging to Bastaushy and "x" denotes stones belonging to Kostaushy. Uppercase letters ("O" and "X") represent ten stones each to reduce board size. This notation helps in visualizing the game state more compactly while maintaining clarity.

For ease of play with a human opponent, the number of stones and the game hole numbers are displayed on the edges of the board.

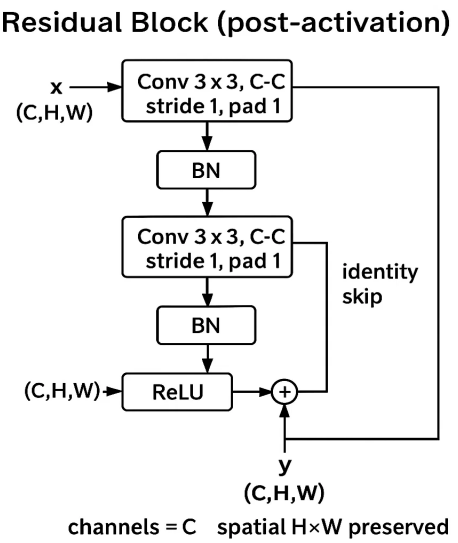
Figure 5
Representation of the board in the starting position

```
Turn  1 Player  1
============================================
000|0|--------------------------------|0|000
009|1|ooooooooo-------------xxxxxxxxx|1|009
009|2|ooooooooo-------------xxxxxxxxx|2|009
009|3|ooooooooo-------------xxxxxxxxx|3|009
009|4|ooooooooo-------------xxxxxxxxx|4|009
009|5|ooooooooo-------------xxxxxxxxx|5|009
009|6|ooooooooo-------------xxxxxxxxx|6|009
009|7|ooooooooo-------------xxxxxxxxx|7|009
009|8|ooooooooo-------------xxxxxxxxx|8|009
009|9|ooooooooo-------------xxxxxxxxx|9|009
============================================
Full board stones: 162.0
```

When Tuzdyk is captured, the cell is marked with a symbol representing the corresponding player. For example, in Figure 7, the Kostaushy has captured cell number 3.

During training, the board is presented as a 2D image of dimension boardx ×boardy (10 × 32) with one channel. The neural network follows the AlphaZero framework, utilizing a convolutional backbone with residual blocks and separate policy and value heads. During feature extraction, the input board state is processed by a convolution with batch normalization and ReLU activation. A sequence of residual blocks captures complex board dynamics.

Figure 6
Basic residual block used in the MCTS network



Each block shown in Figure 6 consists of:

- two convolutional layers with batch normalization;
- a skip connection adding the input to the output before ReLU activation.

The model features a dual-head architecture with separate outputs for policy and value estimation. Outputs action probabilities via:

- a convolution (128 channels) with batch normalization and ReLU;
- a fully connected layer mapping to the action space;
- log softmax activation for move probabilities.

Predicts board evaluation through:

- a convolution (64 channels) with batch normalization and ReLU;
- a fully connected layer (256 neurons);
- a final layer with tanh activation producing a value head.

This architecture balances expressiveness and efficiency, ensuring stable learning through residual blocks and separate optimization for policy and value functions.

The results obtained are described in Subsection 6.2.

Figure 7
Representation of the board with Tuzdyk won by the Kostaushy

```
Turn  51 Player  1
============================================
026|0|OOooooooo-----------------xxxXX|0|023
003|1|ooo-------------------------xxx|1|003
008|2|ooooooooo---------------------x|2|001
000|3|x-----------------------xxxxxx|3|006
001|4|o-------------------------xxxxx|4|005
002|5|oo-----------------------------xx|5|002
007|6|ooooooo--------------------xxxx|6|004
005|7|ooooo-----------------xxxxxxxXX|7|027
005|8|ooooo-----------------xxxxxxxxx|8|009
002|9|oo-------------------------xxxXX|9|023
============================================
Full board stones: 162.0
```

During training AlphaZero-style self-play pipeline is used. MCTS generates training tuples (s_t, π_t, z) where s_t is the board state, π_t is the MCTS policy (visit-count distribution) at step t, and $z \in \{-1, 0, 1\}$ is the game outcome from the current player's perspective. All tuples are stored in a replay buffer with a fixed capacity of 200 000 examples. When the buffer is full, the oldest examples are evicted.

During each training phase, mini-batches are drawn uniformly at random from the buffer (no prioritization). We also persist previously collected examples across runs to ensure continuity and reproducibility. Exploration follows the standard temperature schedule: MCTS policies are sampled stochastically up to move 20 and greedily thereafter. Search strength is controlled by 300 simulations per move.

Model selection is performed via an arena of 40 games between the new checkpoint and the incumbent, with each game starting from a randomized legal position. The challenger replaces the incumbent only if it wins at least 60% of games. Under this protocol, successive checkpoints consistently outperform earlier versions, evidencing a steady increase in playing strength. More details can be found in [22].

5. Development of tools

MARL is a diverse and highly active research area. With the introduction of deep learning in MARL, interest in these problems has grown significantly. As a result, publications and conference reports on this topic indicate the rapid development of the field.

Therefore, MARL researchers also require tools for conducting experiments and reproducing theoretical developments. Recently, numerous frameworks and interfaces have been developed for creating single- and multi-agent environments of various types. Naturally, there are also libraries for training agents, which are especially popular among robotics specialists, as RL is one of the most effective tools for training robots to perform specific tasks.

The source code is written using the frameworks and libraries described in this section and is available at [22].

5.1 Framework for creating an environment

PettingZoo [17] is a simple Pythonic interface designed to represent common RL and MARL problems. PettingZoo includes a wide range of reference environments, useful utilities, and tools to create custom user environments.

5.2 Framework for training agents

Tianshou [23] is a RL framework based on PyTorch. Tianshou provides a high-speed environment and a Pythonic API to build a deep reinforcement learning agent.

It is important to note that not all frameworks support MARL. For example, CleanRL [24], which is widely used in RL, is difficult to adapt for multi-agent environments. However, RLlib [25] is an excellent library for training agents in both single- and multi-agent environments, providing a wide selection of prebuilt algorithms and tools for customizing the training process.

5.3 Library for training an agent based on the MCTS algorithm

To train an AI to play the Togyzkumalak game, the Monte Carlo Tree Search algorithm was employed, which was previously used to train AlphaZero. The environment and training code were developed using the [26] library.

The implementation of the algorithm in this library is easily adaptable to any competitive board game. It is highly flexible and straightforward to implement self-contained reinforcement learning [27]. The library is designed for ease of use in any competitive two-player game and allows for building various DL neural network architecture.

6. Results

This section presents the main results of agent training using DQN and Monte-Carlo Tree Search algorithms. Although the two approaches have differences in the implementation of the environment, the rules of the game embedded in the environment remain consistent.

6.1 Results of Deep Q-Learning

For deep learning, it is necessary to select the neural network architecture, which is a non-trivial task as the success of training depends on it. In the study, the selection criterion was based on the results of the training of various architectures against a random policy. After training, all models played matches against this random policy, and the number of wins was a key factor in choosing the main architecture.

Here are the training details and the main characteristics of the computer resources:

- maximum number of epochs: 2000,
- number of steps per epoch: 1000,
- number of steps for testing: 50,
- batch size: 256,
- discount factor: 0.99.

The Adam optimizer with a learning coefficient of $lr = 3e - 04$ showed the highest results in obtaining a reward.

Details of the training process:

- number of hours: 65,

- GPU: P100 (Kaggle),
- CPU and GPU RAM: 1.5 gb and 243 mb.

The rewards statistics during training for the example of four architectures are shown in Figure 8. Analyzing the graph data, it can be stated that the efficiency of agent training is achieved with the selected training parameters.

Here, vertically, is the average reward value and horizontally, is the step number in each epoch.

To assess out-of-sample playing strength, we use two metrics: win rate against a random policy and win rate against the built-in opponent from The Togyz Qumalaq app at difficulty levels 1–4.

During testing, agents trained using the DQN algorithm were able to defeat only level 1 and level 2 computer opponents out of 4 difficulty levels. The trained policy exhibited short-sighted behavior, focusing on collecting as many stones as possible without considering the defense of game holes.

Figure 8
Dynamics of change in received rewards during training against random policy

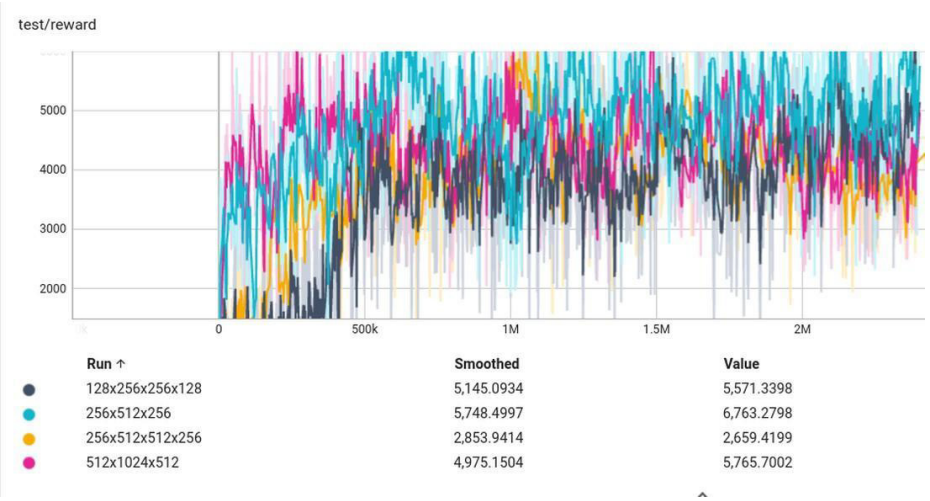
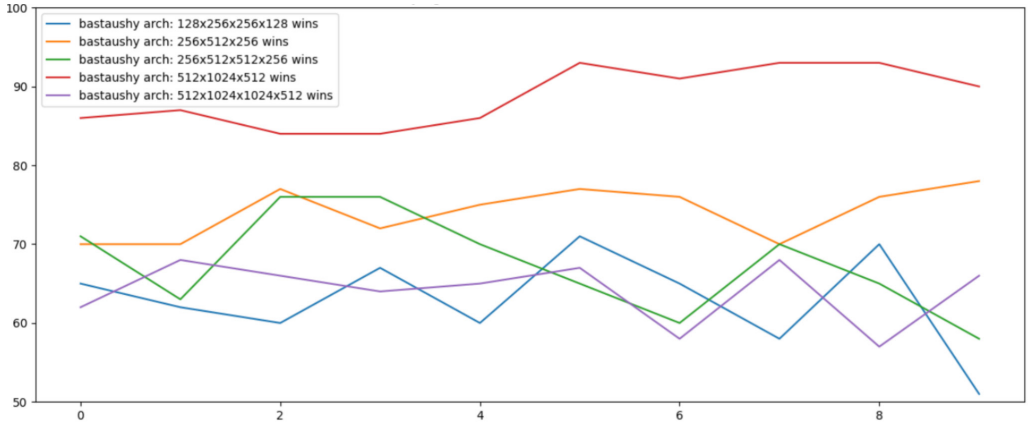


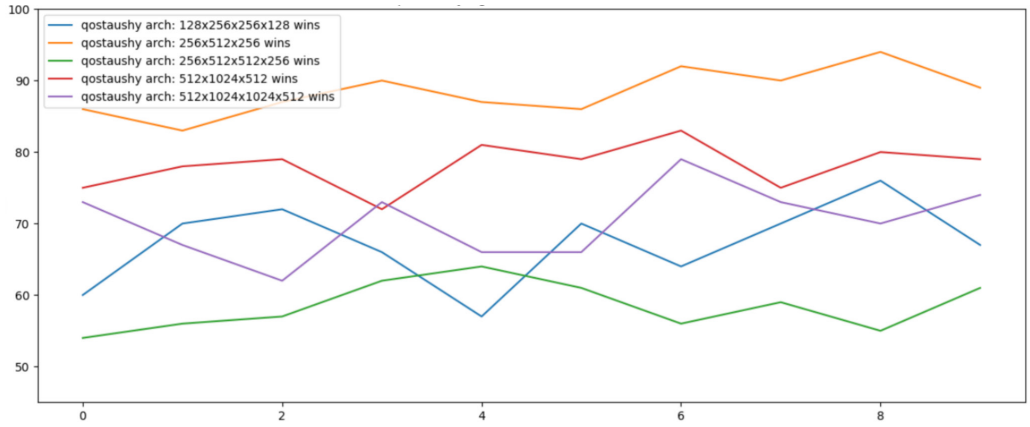
Figure 9 shows win rates of trained agents with five network architectures evaluated against a random policy, separately for the first move (Bastaushy) and the second move (Qostaushy). The 512–1024–512 MLP consistently dominates: as Bastaushy it stays around the 88–93% band across evaluation rounds (Fig. 9), and as Qostaushy it remains in the 78–85% band with peaks above 90% (Fig. 10).

Figure 9
Bastaushy move. Evolution of win rate against a random opponent for different architectures



Smaller models (e.g., 128–256–256–128 or 256–512–256) plateau between ~60% and ~78%. These results guided this choice of 512–1024–512 as the default architecture for subsequent experiments.

Figure 10
Qostaushy move. Evolution of win rate against a random opponent for the same architectures



6.2 Results of Monte Carlo Tree Search algorithm

Overcoming challenges described in Section 3 required finding a suitable representation of the board and constructing a customized neural network architecture for the algorithm.

The neural network architecture is described in Subsection 4.2. We will provide the training details and describe the implementation using the self-play technique.

During the experiments, the optimal number of residual blocks was found to be 30. One of the key training parameters was numMCTSSims, which determines the number of game moves simulated by MCTS. A trade-off study between execution time and stable learning through self-play resulted in a value of 300. A fixed learning rate did not yield good results, so the training utilized a CosineAnnealingLR scheduler and a momentum SGD optimizer. More details on the training process can be found in [22].

Main training parameters:

- number of channels in layers: 256,
- number of residual blocks: 30,
- learning rate: 0.001 → 0.0001,
- dropout probability: 0.3,
- number of epochs: 20,
- mini-batch size: 64.

The network consists of an input stem Conv3×3 (1→256) + BN, a stack of 30 residual blocks, a policy head Conv1×1 (256→128) + BN + FC ($128 \times board_x \times board_y \rightarrow 9$), and a value head Conv1×1 (256→64) + BN + FC ($64 \times board_x \times board_y \rightarrow 256$) + FC ($256 \rightarrow 1$). Here, BN denotes Batch Normalization and FC denotes a fully connected (linear) layer with bias. The model contains 41,100,362 trainable parameters (weights + biases + BN affine).

During training, the open source library [27] was used, which is available at [26]; more details are given in Subsection 5.3. After training, the model plays several matches against the previously version of itself; if > 60% wins, a new model is accepted; otherwise, it is rejected.

Details of the training process:

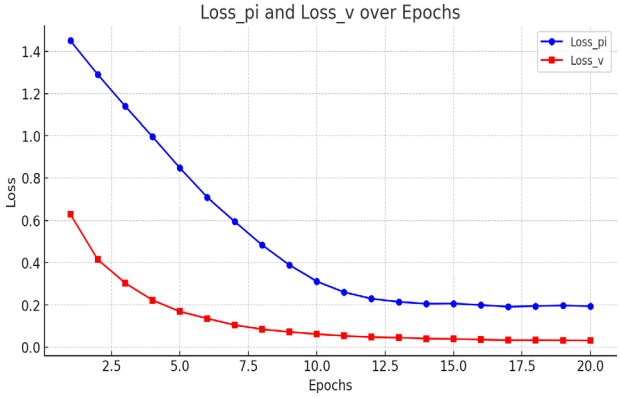
- number of hours: 172,
- GPU: GeForce RTX 4060 Max-Q / Mobile,
- CPU and GPU RAM: 16 gb and 8 gb.

The dynamics of dual head loss during most iterations followed a consistent pattern as in Figure 9. Trained agents can still win against novices and a random policy; however, to outperform professional players, more computing resources are required for training and experimentation. Unfortunately, there was insufficient memory and processing power for a wider or deeper network, which could be crucial to defeating a professional player.

However, the training dynamics in the first dozens of iterations indicated that this neural network has the potential to become a champion in Togyzkumalak. As the number of training examples increased, the agent consistently outperformed its previous versions, demonstrating a steady improvement in its playing strength.

Training dynamics of the MCTS policy – value network is shown in Figure 11. Policy loss (blue; cross-entropy to MCTS visit distribution) and value loss (red; MSE to game outcome) versus training epoch. The value loss converges quickly and stabilizes near 0.04; the policy loss decreases steadily and plateaus around 0.19 after ≈14 epochs, indicating overall convergence of the network.

Figure 11
Dual-head loss dynamics during one iteration



To assess how the convolutional policy – value network used with MCTS translates into practical playing strength, we evaluated the final checkpoint against The Togyz Qumalaq mobile app. The MCTS agent consistently defeated levels 1–3 (as Bastaushy), while level 4 remains an open challenge. For context, the best DQN baseline reliably beats only levels 1–2, reflecting the limits of a flat MLP without spatial inductive bias.

7. Discussion and future work

The main goal of this work was to develop trained agents that play the Togyzkumalak game effectively. To achieve this, a multi-agent environment was prepared, and research work was carried out to design a reward system, select the appropriate architecture, and choose a learning algorithm.

In addition to the algorithms described in Section 4, experiments were carried out with other methods, such as Proximal Policy Optimization and Rainbow. However, based on the comparison of results from these experiments, the Deep Q-Learning and Monte Carlo Tree Search algorithms were selected.

While DQN is effective in many control and Atari-like benchmarks, it is not ideal for combinatorial board games with sparse rewards and long-horizon tactics. Accordingly, we treat DQN as a baseline and adopt a planning-based approach (policy-value networks with MCTS-guided self-play), which enables the model to discover board patterns and apply state-specific policies with deep lookahead from each position.

The AlphaZero algorithm uses information about the game's specifics. For example, the main property of all the listed board games (Go, Chess, Shogi and Togyzkumalak) is symmetry, i.e. the board can be symmetrically reflected along the main axis; in the case of the Togyzkumalak game, this is the vertical axis. In future research, there is interest in applying the methods used in the training of MuZero AI [16].

This research achieved promising results in practical applications.

Acknowledgments

The author would like to thank the Institute of Mathematics and Mathematical Modeling for their technical and organizational support during this study.

Funding

This research has been funded by the Science Committee of The Ministry of Science and Higher Education of the Republic of Kazakhstan (Grant No. AP23488302).

Conflicts of Interest

The author declares no conflict of interest.

References

1. R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, 2nd ed., ser. Adaptive Computation and Machine Learning. Cambridge, MA, USA: MIT Press, 2018.
2. S. V. Albrecht, F. Christianos, and L. Schäfer, *Multi-Agent Reinforcement Learning: Foundations and Modern Approaches*. Cambridge, MA, USA: MIT Press, 2024.
3. K. Zhang, Z. Yang, and T. Başar, "Policy optimization provably converges to Nash equilibria in zero-sum linear quadratic games," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 32, 2019.

4. C. Daskalakis, D. J. Foster, and N. Golowich, "Independent policy gradient methods for competitive reinforcement learning," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 33, 2020, pp. 5527–5540.

5. C.-Y. Wei, C.-W. Lee, M. Zhang, and H. Luo, "Last-iterate convergence of decentralized optimistic gradient descent/ascent in infinite-horizon competitive Markov games," in *Proc. 34th Conf. Learning Theory (COLT)*, ser. Proceedings of Machine Learning Research, vol. 134, M. Belkin and S. Kpotufe, Eds., Aug. 2021, pp. 4259–4299.

6. J. von Neumann, "Zur Theorie der Gesellschaftsspiele," *Mathematische Annalen*, vol. 100, no. 3, pp. 295–320, 1928.

7. J. von Neumann, O. Morgenstern, and A. Rubinstein, *Theory of Games and Economic Behavior* (60th Anniversary Commemorative Edition). Princeton, NJ, USA: Princeton University Press, 1944.

8. R. J. Vanderbei, *Linear Programming: Foundations and Extensions*. Cham, Switzerland: Springer, 2020.

9. M. Bowling and M. Veloso, "Multiagent learning using a variable learning rate," *Artificial Intelligence*, vol. 136, no. 2, pp. 215–250, Apr. 2002.

10. B. Banerjee and J. Peng, "Performance bounded reinforcement learning in strategic interactions," in *Proc. 19th National Conf. Artificial Intelligence (AAAI)*, San Jose, CA, USA, 2004, pp. 2–7.

11. R. Powers and Y. Shoham, "Learning against opponents with bounded memory," in *Proc. 19th Int. Joint Conf. Artificial Intelligence (IJCAI)*, San Francisco, CA, USA, 2005, pp. 817–822.

12. D. Silver, J. Schrittwieser, K. Simonyan, *et al.*, "Mastering the game of Go without human knowledge," *Nature*, vol. 550, pp. 354–359, 2017.

13. T. Bansal, J. Pachocki, S. Sidor, I. Sutskever, and I. Mordatch, "Emergent complexity via multi-agent competition," in *Proc. Int. Conf. Learning Representations (ICLR)*, 2018.

14. E. Todorov, T. Erez, and Y. Tassa, "MuJoCo: A physics engine for model-based control," in *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS)*, 2012, pp. 5026–5033.

15. C. Berner, G. Brockman, B. Chan, *et al.*, "Dota 2 with large scale deep reinforcement learning," *arXiv preprint arXiv:1912.06680*, 2019.

16. J. Schrittwieser, I. Antonoglou, T. Hubert, *et al.*, "Mastering Atari, Go, chess and shogi by planning with a learned model," *Nature*, vol. 588, pp. 604–609, 2020.

17. J. K. Terry, B. Black, A. Hari, *et al.*, "PettingZoo: Gym for multi-agent reinforcement learning," *arXiv preprint arXiv:2009.14471*, 2020.

18. V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. A. Riedmiller, "Playing Atari with deep reinforcement learning," *arXiv preprint arXiv:1312.5602*, 2013.

19. H. van Hasselt, "Double Q-learning," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 23, 2010.

20. D. Silver, T. Hubert, J. Schrittwieser, *et al.*, "Mastering chess and shogi by self-play with a general reinforcement learning algorithm," *arXiv preprint arXiv:1712.01815*, 2017.

21. T. Romstad, M. Costalba, and J. Kiiski. "Stockfish chess engine." Accessed: Mar. 27, 2025. [Online]. Available: https://stockfishchess.org/

22. zhus-dika. "Togyzkumalak agent." Accessed: Mar. 27, 2025. [Online]. Available: https://github.com/zhus-dika/togyz-qumalaq-agent

23. Tianshou Team. "Tianshou: A library for deep reinforcement learning in PyTorch." Accessed: Mar. 27, 2025. [Online]. Available: https://tianshou.org/en/stable/index.html

24. S. Huang, R. F. J. Dossa, C. Ye, J. Braga, D. Chakraborty, K. Mehta, and J. G. M. Araújo, "CleanRL: High-quality single-file implementations of deep reinforcement learning algorithms," *Journal of Machine Learning Research*, vol. 23, no. 274, pp. 1–18, 2022.

25. Ray. "RLlib: Scalable reinforcement learning." Accessed: Mar. 27, 2025. [Online]. Available: https://docs.ray.io/en/latest/index.html

26. S. Nair. "Alpha-Zero-General implementation." Accessed: Mar. 27, 2025. [Online]. Available: https://github.com/suragnair/alpha-zero-general

27. S. Thakoor, S. Nair, and M. Jhunjhunwala, "Learning to play Othello without human knowledge," Stanford Univ., Stanford, CA, USA, Tech. Rep., 2016.

Information about the authors:

Dinara Zhussupova – PhD. Dr. Dinara Zhussupova is a Leading Researcher at the Institute of Mathematics and Mathematical Modeling. She received her PhD in Mathematics from L.N. Gumilyov Eurasian National University. Her academic and research work focuses on mathematical modeling, numerical methods, artificial intelligence, and reinforcement learning. She has experience in interdisciplinary research at the intersection of mathematics, computer science, and intelligent systems (Almaty, Kazakhstan, e-mail: zhus.dinara@gmail.com. ORCID iD: 0000-0002-1860-0843).

Submission received: 10 April, 2025
Revised: 17 February, 2026
Accepted: 17 February, 2026

IRSTI 28.23.27 https://doi.org/10.26577/jpcsit4220262

Abdulla Sapargali [1*], Khansa Arshad [2], Muhammad Ilyas [1]

[1]Kazakh British Technical University, Almaty, Kazakhstan
[2]Government College University, Lahore, Pakistan
*e-mail: abdulla.sapargali.2002@gmail.com

PRE-FILTERING OF GRID MAPS WITH SKELETONIZATION AND MORPHOLOGY METHOD FOR EFFICIENT PATH PLANNING FOR MOBILE ROBOTICS

Abstract. SLAM-based occupancy grid maps are widely used for indoor mobile robots, but their noise, jagged obstacle borders, and spurious narrow gaps can mislead the classical grid-based path planners, e.g. A* algorithm, causing unstable routes, edge-hugging behavior, and increased computation. This manuscript proposes a lightweight pipeline to improve global path planning robustness and speed in warehouse-like environments. The approach first converts the SLAM map into a conservative binary representation and applies morphological refinement to remove small artifacts and smooth free-space connectivity. Next, the free space is skeletonized and transformed into a graph that captures the topology of navigable corridors. Global path planning is then performed using a hybrid strategy: a local grid search connects the start and goal regions to the skeleton, while the main route is computed on the skeleton graph. Experiments in two warehouse scenarios show that the proposed method substantially reduces node expansions and path planning time compared to full-grid A* algorithm, while producing more physically plausible paths that avoid false passages in noisy maps. These results indicate that morphological and skeleton-based global path planning can serve as a practical solution to enhance efficiency for robust navigation in realistic SLAM based path planning.

Keywords: Warehouse Robotics, SLAM, Occupancy Grid Map, Morphological Filtering, Skeletonization, Graph-Based Path Planning, A* Search.

1. Introduction

Autonomous mobile robots (AMRs) and mobile manipulators are becoming core components of warehouse intralogistics, where robots must navigate narrow aisles efficiently and reliably under limited onboard computation. Recent surveys and industry-oriented studies highlight that, beyond the choice of controller, the quality of the environmental representation and the efficiency of global path planning strongly influence overall system robustness and throughput in real warehouse deployments [1]–[4].

In many ROS-based indoor systems, SLAM-generated 2D occupancy grids remain a practical standard, but they often contain artifacts such as jagged obstacle borders, small spurious gaps, and isolated noisy cells [5]–[7]. These defects are problematic for classical grid-based A* path planning: the search may exploit false connectivity through one-cell passages, generate edge-hugging or irregular routes, and expand many unnecessary nodes, increasing planning time—especially in larger maps or higher resolutions. To address this, this paper proposes a lightweight path planning pipeline for warehouse maps: (i) conservative binarization of the occupancy grid followed by morphological refinement to remove small defects and enforce safe clearance, inspired by morphology-based planning practices [8]; (ii) skeletonization of refined free space and conversion to a compact graph representation for corridor-centered routing [9], [10]; and (iii) a hybrid planning strategy that connects start/goal locally on the grid while computing the main route on the skeleton graph. The goal is to reduce search effort and improve path plausibility in the presence of typical SLAM map noise.

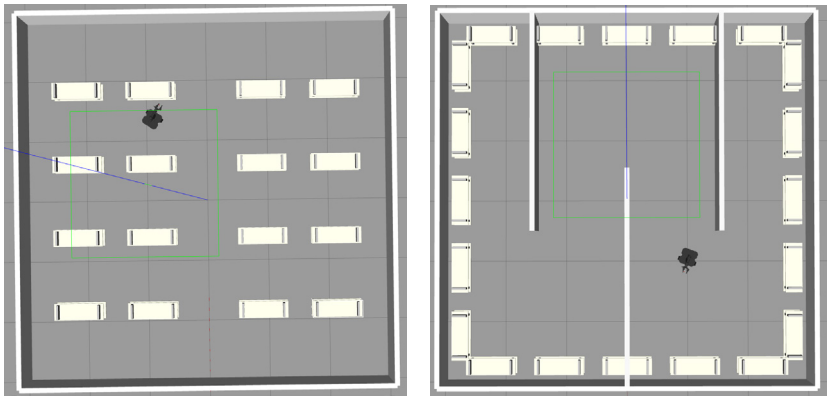
2. Materials and methods

2.1 Experimental Design and Setup

The study compares two global path planning approaches on 2D SLAM occupancy maps: (i) baseline full grid A* and (ii) the proposed hybrid grid–skeleton planner. Experiments were executed in a reproducible ROS/Gazebo simulation pipeline, using standard ROS map messages and a fixed start–goal configuration per scenario [11],

[12], [13]. Experiment will consider 2 cases, where Case 1 without any walls, only shelves and Case 2, where walls included in map A* was selected as the baseline due to its widespread use in robotics navigation as an optimal heuristic search on graphs [14].

Figure 1
Two compared scenario: (a) Case 1; (b) Case 2



(a) (b)

2.2 Map Input and Conservative Binarization

The input map is a SLAM-produced 2D occupancy grid, a common representation for indoor navigation and path planning [5], [8]–[10]. Each cell is assigned to free, occupied, or unknown. To prevent path planning through uncertain "micro-gaps" and partially observed regions, the grid is converted into a conservative binary map: only confidently free cells are kept as traversable, while occupied and unknown cells are treated as obstacles. This choice prioritizes physical feasibility over optimistic connectivity, which is important when SLAM artifacts are present [5], [8].

2.3 Morphological Refinement of the Binary Map

To reduce the impact of jagged borders, isolated noisy pixels, and spurious narrow connections, the binary map is refined using morphological operators. Specifically:

a) Closing is applied to remove small holes and bridge tiny gaps that are not reliably traversable.

b) Dilation is then applied to introduce a safety margin approximating the robot footprint in grid cells.

Morphological preprocessing is a practical technique for improving the robustness of path planning on occupancy maps without modifying the SLAM backend [15]. Structuring element sizes are selected in grid cells according to map resolution and robot dimensions (i.e., the dilation radius corresponds to the desired obstacle clearance).

2.4 Skeletonization and Skeleton Graph Construction

From the refined free-space mask, a 1-cell-wide skeleton is extracted via thinning/skeletonization to approximate corridor centerlines and preserve the topology of navigable space [16], [17]. Thinning can be implemented using classic fast parallel methods (e.g., Zhang–Suen-type thinning), which are commonly used for skeleton extraction [18].

The skeleton image is converted into a weighted graph G=(V,E):

a) Vertices correspond to junctions and endpoints (pixels with degree ≠2 in 8-neighborhood connectivity).

b) Edges are formed by chains of degree-2 pixels between vertices, with edge weights equal to the geometric length of the chain.

To reduce redundancy, intermediate degree-2 points are compressed into single edges, producing a compact graph suitable for efficient global search [16], [17].

2.5 Path Planning Methods

Baseline (Full grid A*):

A* searches directly on the (refined) binary grid, using 8-connected moves and edge costs proportional to Euclidean step length. The heuristic is the

Euclidean distance to the goal, ensuring admissibility under standard conditions [14].

Proposed (Hybrid grid–skeleton planning):

Planning is performed in three stages:

a) Local grid A* from start to the nearest reachable skeleton vertex,

b) Graph A* on the skeleton graph between skeleton vertices,

c) Local grid A* from the skeleton to the goal.

This design keeps endpoint flexibility (grid search near start/goal) while leveraging a compact topological representation for the main route to reduce the explored state space [16], [17]. The final path is assembled by concatenating the three segments; optional post-processing (e.g., simple shortcutting) can be applied if needed, but the reported results use the direct concatenation to keep comparisons consistent.

2.6 Evaluation Metrics

For each scenario, the following metrics are recorded:

a) Expanded nodes (search effort) for both the grid and graph searches,

b) Planning runtime (ms) measured as wall-clock time for the planning call,

c) Path length (in grid units/pixels) and number of path nodes to characterize the resulting route.

These metrics are standard and directly reflect computational efficiency and route geometry in graph-search path planning on occupancy maps [14], [16].

2.7 Replicability Notes

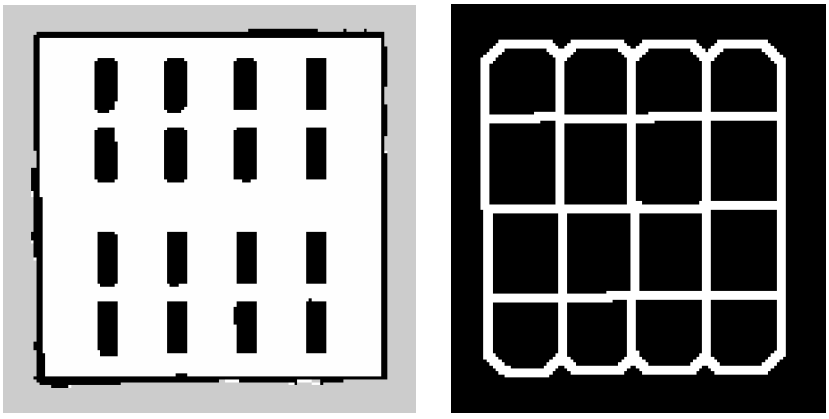
The full pipeline is reproducible given: (i) an occupancy grid map (ROS nav_ msgs/OccupancyGrid), (ii) robot footprint parameters to set the dilation/clearance, (iii) start and goal coordinates in map frame, and (iv) fixed neighborhood/cost definitions for A*. The morphology and skeleton steps are deterministic for fixed parameters, and the hybrid planner uses the same A* implementation as the baseline for fair comparison.

3. Results

The proposed pipeline was evaluated on two tested maps: Case 1 and Case 2. For each case, we first considered the original occupancy representation (without skeletonization) and then the corresponding skeleton representation used for graph-based routing (Figures 2 and 3). In both cases, the skeleton captures the main navigable corridors as a compact structure for global path planning.

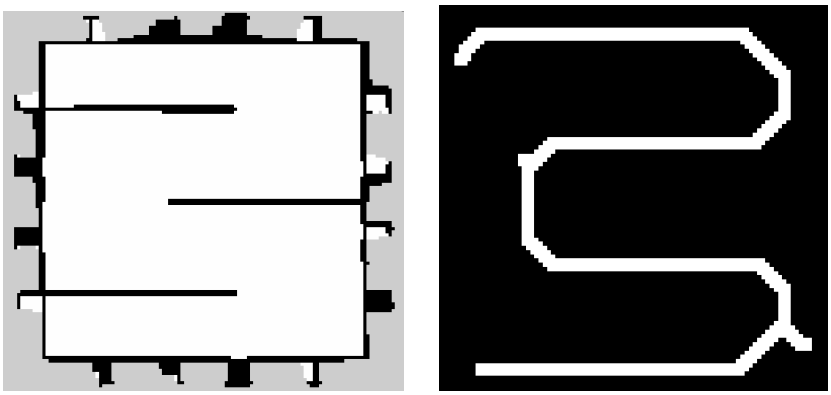
Figure 1 presents the overall real cases that were used in experiments. Figures 4 and 5 show example planned routes for the baseline full grid A* and the proposed hybrid method in Case 1 and Case 2, respectively.

Figure 2
Tested map Case 1: (a) Without Skeletonization; (b) After Skeletonization



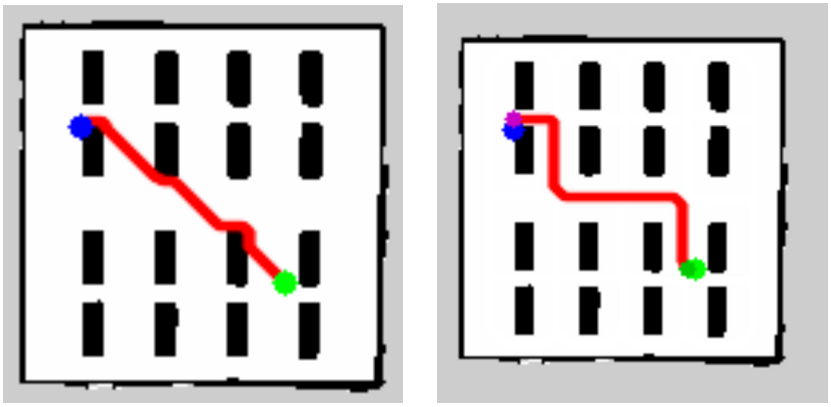
(a) (b)

Figure 3
Tested map Case 2: (a) Without Skeletonization; (b) After Skeletonization



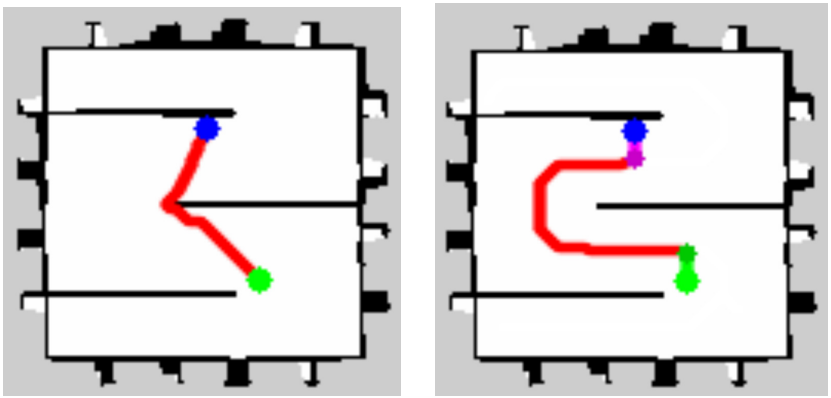
(a) (b)

Figure 4
Tested map Case 1: (a) A without skeletonization; (b) A* with skeletonization*



(a) (b)

Figure 5
Tested map Case 2: (a) A without skeletonization; (b) A* with skeletonization*



(a) (b)

Quantitative results are summarized in Table 1. For Case 1, the baseline full grid A* expanded 1840 nodes with a planning time of 23.12 ms, while the hybrid method expanded 1005 nodes with 3.75 ms runtime. The resulting path length was 95.3 px (baseline) and 122.1 px (hybrid), with 76 and 119 path nodes, respectively.

For Case 2, the baseline full grid A* expanded 1365 nodes with a planning time of 16.19 ms, while the hybrid method expanded 498 nodes with 1.70 ms runtime. The resulting path length was 70.3 px (baseline) and 118.6 px (hybrid), with 56 and 113 path nodes, respectively.

Table 1
Comparison of full grid-based A and hybrid skeleton-based path planning (Cases 1 and 2)*

Metric	Case 1 – Full A*	Case 1 – Hybrid	Case 2 – Full A*	Case 2 – Hybrid
Expanded nodes	1840	1005	1365	498
Runtime (ms)	23.12	3.75	16.19	1.70
Path length (pixels)	95.3	122.1	70.3	118.6
Number of path nodes	76	119	56	113

4. Discussion

The results demonstrate that the proposed hybrid grid–skeleton approach consistently improves path planning efficiency compared to full grid A*. In both scenarios, the hybrid method reduces expanded nodes and runtime, indicating that most of the search is shifted from the dense grid to a compact topological representation of free space. This behavior is aligned with the motivation of using skeleton/graph structures to decrease the number of explored states while preserving connectivity for global routing [16], [17].

A key practical benefit is robustness to typical SLAM occupancy-grid artifacts. Occupancy grids frequently contain jagged borders, isolated noisy cells, and thin "false" passages created by sensor noise or partial observations [5], [8]–[10]. The conservative binarization and morphological refinement step reduces these defects, and the skeleton then captures the dominant corridor structure of the refined free space. In Case 2, where artifacts and complex connectivity are more likely, the hybrid method achieves the strongest reduction in expanded nodes, suggesting that the pipeline is particularly helpful when raw grid connectivity is unreliable.

The observed trade-off is an increase in path length (in pixels) for the hybrid method in both cases. This increase is expected for two reasons. First, morphological dilation introduces safety clearance that removes narrow shortcuts and pushes routes away from obstacle boundaries, favoring physically feasible trajectories over optimistic grid connectivity [15]. Second, skeleton-based routing tends to follow corridor centerlines, which may be slightly longer than a shortest path that cuts corners or uses marginal gaps. For warehouse navigation, this trade-off can be acceptable or even desirable because centerline routes improve clearance and reduce collision risk in narrow aisles, supporting stable execution by downstream local controllers.

Several limitations should be noted. The quality of the final route depends on parameter choices for morphological refinement (e.g., structuring element radius). If the dilation is too aggressive, it may over-constrain free space and unnecessarily lengthen paths or disconnect corridors; if too small, spurious gaps may remain. In addition, the current evaluation focuses on static maps and fixed start–goal pairs; dynamic obstacles and frequent map updates are handled primarily by local planners in standard navigation stacks rather than by the global planner. Future work can therefore include automated parameter tuning based on robot footprint and map resolution, incremental skeleton updates as the map changes, and evaluation on a larger set of start–goal pairs and more diverse warehouse layouts.

Overall, the findings indicate that combining lightweight morphological preprocessing with skeleton-graph routing is an effective and practical enhancement for global path planning on SLAM occupancy grids, especially in environments where grid artifacts can mislead full-grid A* search.

5. Conclusions

This paper presented a lightweight global path planning pipeline for warehouse-like SLAM occupancy grid maps that combines conservative binarization, morphological map refinement, free-space skeletonization, and a hybrid grid–skeleton A* strategy. Across two scenarios, the hybrid approach reduced search effort (expanded nodes) and planning runtime compared to full grid A*, indicating more efficient global path planning on noisy or artifact-prone maps. Although the resulting routes were longer in pixel length, they were biased toward corridor-centered, clearance-aware paths, which are typically more physically plausible for navigation in narrow aisles. Future work will focus on automatic selection of morphological parameters for different map resolutions and robot footprints, incremental skeleton updates for online mapping, and evaluation on larger sets of start–goal pairs and dynamic warehouse conditions.

Acknowledgments

This work is supported by School of IT and Engineering (SiTE), Kazakh-British Technical University (KBTU, Kazakhstan.

Funding

This research received no external funding.

Conflicts of Interest

The authors declare no conflict of interest.

Author Contributions

Conceptualization, A.S. and M.I.; Methodology, A.S.; Software, A.S.; Validation, A.S., K.A. and M.I.; Formal Analysis, K.A.; Investigation, A.S.; Resources, M.I.; Data Curation, K.A.; Writing – Original Draft Preparation, A.S.; Writing – Review & Editing, K.A. and M.I.; Visualization, A.S.; Supervision, M.I.; Project Administration, M.I.

All authors have read and agreed to the published version of the manuscript.

References

1. A. K. Grover and M. H. Ashraf, "Autonomous mobile robots for warehousing and distribution industry: A step toward intralogistics 4.0," in *Digitization in Supply Chain Management: Trends, Challenges and Solutions*, S. Yeniyurt and S. Carnovale, Eds. Singapore: World Scientific, 2024, pp. 153–183, doi: 10.1142/9789811286636_0008.

2. G. Fragapane, D. Ivanov, M. Peron, F. Sgarbossa, and J. O. Strandhagen, "Increasing flexibility and productivity in Industry 4.0 production networks with autonomous mobile robots and smart intralogistics," *Annals of Operations Research*, vol. 308, no. 1, pp. 125–143, 2022, doi: 10.1007/s10479-020-03526-7.

3. T. Lackner, J. Hermann, C. Kuhn, and D. Palm, "Review of autonomous mobile robots in intralogistics: State-of-the-art, limitations and research gaps," *Procedia CIRP*, vol. 130, pp. 930–935, 2024, doi: 10.1016/j.procir.2024.10.187.

4. R. Keith and H. M. La, "Review of autonomous mobile robots for the warehouse environment," *arXiv preprint arXiv:2406.08333*, 2024, doi: 10.48550/arXiv.2406.08333.

5. D. Damjanović, P. Biočić, S. Prakljačić, D. Činčurak, and J. Balen, "A comprehensive survey on SLAM and machine learning approaches for indoor autonomous navigation of mobile robots," *Machine Vision and Applications*, vol. 36, no. 3, p. 55, 2025, doi: 10.1007/s00138-025-01673-0.

6. J. A. Placed, J. Strader, H. Carrillo, N. Atanasov, V. Indelman, L. Carlone, and J. A. Castellanos, "A survey on active simultaneous localization and mapping: State of the art and new frontiers," *IEEE Trans. Robot.*, vol. 39, no. 3, pp. 1686–1705, 2023, doi: 10.1109/TRO.2023.3248510.

7. I. Abaspur Kazerouni, L. Fitzgerald, G. Dooly, and D. Toal, "A survey of state-of-the-art on visual SLAM," *Expert Systems with Applications*, vol. 205, p. 117734, 2022, doi: 10.1016/j.eswa.2022.117734.

8. B. B. K. Ayawli, A. Y. Appiah, I. K. Nti, F. Kyeremeh, and E. I. Ayawli, "Path planning for mobile robots using Morphological Dilation Voronoi Diagram Roadmap algorithm," *Scientific African*, vol. 12, art. e00745, 2021, doi: 10.1016/j.sciaf.2021.e00745.

9. H. R. Beom and H. S. Cho, "Path planning for mobile robot using skeleton of free space," in *Information Control Problems in Manufacturing Technology 1992*. Oxford, U.K.: Pergamon, 1993, pp. 355–359, doi: 10.1016/B978-0-08-041897-1.50066-3.

10. F. M. Santa, E. J. Gomez, and H. M. Ariza, "Global path planning for mobile robots using image skeletonization," *Indian Journal of Science and Technology*, vol. 10, no. 14, pp. 1–6, 2017, doi: 10.17485/ijst/2017/v10i14/112175.

11. M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "ROS: An open-source Robot Operating System," in *Proc. ICRA Workshop on Open Source Software*, Kobe, Japan, 2009.

12. N. Koenig and A. Howard, "Design and use paradigms for Gazebo, an open-source multi-robot simulator," in *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS)*, 2004, pp. 2149–2154, doi: 10.1109/IROS.2004.1389727.

13. P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Trans. Syst. Sci. Cybern.*, vol. 4, no. 2, pp. 100–107, 1968, doi: 10.1109/TSSC.1968.300136.

14. P. Soille, *Morphological Image Analysis: Principles and Applications*, 2nd ed. Berlin, Germany: Springer, 2003.

15. H. Niu, X. Ji, L. Zhang, F. Wen, R. Ying, and P. Liu, "A skeleton-based topological planner for exploration in complex unknown environments," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*, 2025, pp. 11766–11772, doi: 10.1109/ICRA55743.2025.11128159.

16. G. O. Flores-Aquino, O. Gutierrez-Frias, and J. I. Vasquez, "Path planning using a one-shot-sampling skeleton map," *arXiv preprint arXiv:2507.02328*, 2025, doi: 10.48550/arXiv.2507.02328.

17. T. Y. Zhang and C. Y. Suen, "A fast parallel algorithm for thinning digital patterns," *Communications of the ACM*, vol. 27, no. 3, pp. 236–239, 1984, doi: 10.1145/357994.358023.

Information about the authors:

Abdulla Sapargali – 2nd year Master student at Kazakh-British Technical University (KBTU). His research interest are: Machine learning, robotic, image processing, and data science (ORCID iD: 0000-0003-0295-3449).

Khansa Arshad – Master student in Mathematical modeling at Government College University Lahore (GCUL), Pakistan. Her research interests are: mathematical modelling, machine learning, data analysis, signal processing (ORCID iD: 0009-0005-6212-4118).

Muhammad Ilyas – Ph.D, is a Professor of Robotics and Mechatronics Engineering at Kazakh-British Technical University (KBTU). He received his PhD in Robotics and Virtual Engineering from UST S.Korea in 2016. Dr. Ilyas has over 20 years of experience in electronics, robotics, artificial intelligence and machine learning. His research interests include robotics navigation, deep learning applications in robotics and computer vision (Almaty, Kazakhstan, e-mail: m.ilyas@kbtu.kz. ORCID iD: 0000-0001-7555-3839).

Submission received: 8 January, 2026
Revised: 24 January, 2026
Accepted: 24 January, 2026

IRSTI 28.23.15 https://doi.org/10.26577/jpcsit4220263

Zaki Al-Farabi, Ilyas Umurbekov, Iliyas Tursynbek, Adilet Yesbayev, Zhanibek Rysbek, Almaskhan Baimyshev, Zhanat Kappassov*

Nazarbayev University, Astana, Kazakhstan
*e-mail: zhkappassov@nu.edu.kz

LOW-COST NIGHT VISION CAMERA BASED ON A MODIFIED CONSUMER WEBCAM AND LIGHTWEIGHT REAL-TIME ENHANCEMENT

Abstract. Low-light perception is a core requirement for robotics, inspection, and wearable situational awareness, yet reliable night- vision solutions often require expensive sensors or computationally heavy processing. This manuscript presents (i) a low-cost near- infrared (NIR) night-vision camera system with real-time enhancement and mixed-reality visualization, and (ii) a controlled benchmarking protocol that compares multiple sensor modalities and classical en- hancement pipelines on a Linux embedded platform. To prioritize real-time feasibility, we deliberately avoid neural networks and other AI/ML models, and instead benchmark lightweight classical methods: CLAHE, bilateral filtering + CLAHE, Non-Local Means (NLM) + CLAHE, Retinex SSR, Retinex SSR with percentile normalization, and two proposed pipelines (a CLAHE-based pipeline and a Retinex integrated variant). Experiments were conducted in a fully dark room using identical illumination conditions and multiple observation distances. We report runtime (ms/frame, FPS) and no- reference quality metrics (entropy, edge strength, Laplacian variance, RMS contrast, mean intensity) to accommodate the absence of ground-truth references. A representative run shows that the proposed CLAHE-based pipeline achieves 246.5 FPS at 640×480 while substantially improving objective no-reference metrics relative to raw frames, whereas Retinex with percentile normalization yields higher contrast/entropy but at 12.7 FPS. Hardware benchmarking indicates that thermal imaging provides the clearest visibility but at high cost/power, Kinect requires stronger illumination, and event-based sensing offers extremely low latency but requires temporal modulation to observe static targets.

Keywords: night vision, low-light enhancement, Retinex, CLAHE, embedded vision, Jetson, event camera, thermal camera, HoloLens.

1. Introduction

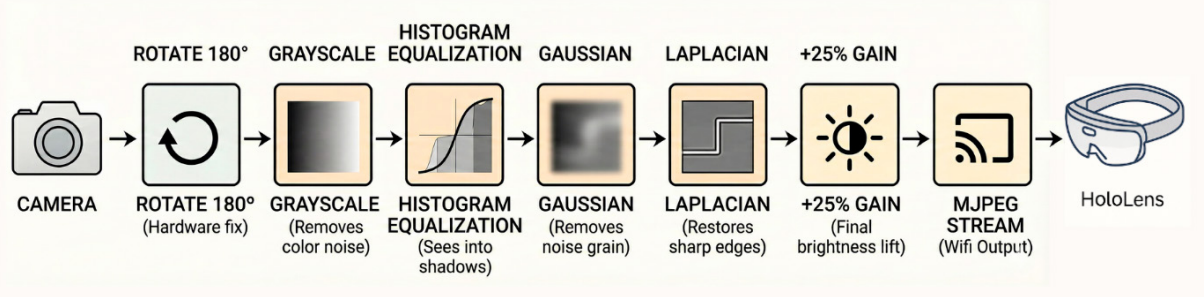
Night-vision sensing enables perception in environments where visible-light cameras fail, including nighttime naviga- tion, surveillance, search-and-rescue, and industrial inspection. Low-light frames commonly exhibit reduced contrast and amplified noise due to sensor gain and exposure control, motivating enhancement methods that improve visibility while preserving structure [1], [2].

Classical enhancement pipelines remain attractive for embedded systems because they are transparent, lightweight, and do not require training data or model deployment. Local contrast enhancement is often performed using adaptive histogram equalization and its variants [3], including contrast-limited adaptive histogram equalization (CLAHE) [4], [5]. Edge- preserving smoothing can be performed using anisotropic diffusion [6] or bilateral filtering [7], while patch-based denoising such as Non-Local Means can yield strong noise suppression at higher computational cost [8]. Illumination normalization is frequently addressed with Retinex theory [9] and practical center/surround Retinex formulations [10]. These classical methods contrast with learning-based low-light enhancement approaches such as LIME [11], RetinexNet [12], and Zero- DCE [13], which can provide strong enhancement but typically increase system complexity and may exceed the intended compute budget for lightweight embedded deployment.

This manuscript benchmarks both hardware and software under identical dark-room conditions. Hardware includes a thermal camera (FLIR T540 class) [14], Xbox Kinect v1 [15], an event-based camera platform (Prophesee Metavision EVK4-HD class) [16]–[18], and our own low-cost NIR camera system. Software benchmarking compares multiple

classical pipelines in real time and reports metrics to support objective comparison imagery. The complete system source code is available publicly [19]. An overview of the real-time enhancement pipeline and MJPEG streaming workflow is shown in Figure 1.

Figure 1
Overview of the real-time enhancement pipeline and MJPEG streaming to HoloLens



2. Materials and methods

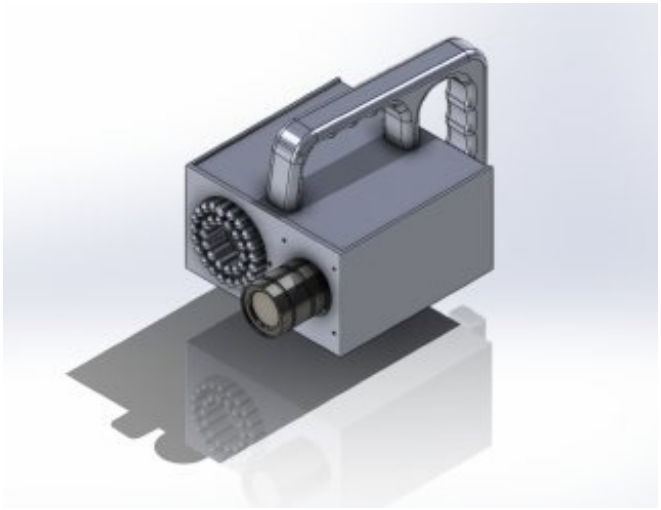
2.1 Embedded Platform and System Implementation

All experiments were executed on an NVIDIA Jetson Orin NX (16GB-class) embedded platform running Linux [20]. The capture-process-record pipeline was implemented in C++ using OpenCV [21], which enables native compilation and access to optimized image processing primitives commonly used in real-time vision systems. The proposed system is designed for low-latency streaming: enhanced frames are encoded and streamed as MJPEG to multiple clients for visualization, including a Unity-based client running on HoloLens 2 [22].

2.2 3D Printed Enclosure

To house the modified camera and IR emitter as a compact unit, a custom enclosure was designed using CAD software and manufactured using a 3D printer. The enclosure provides structural stability, protects internal components, and ensures that the emitter and lens remain fixed and aligned during operation. The handle mounted on the enclosure facilitates handheld operation and improves device portability. Figure 2 shows the CAD rendering of the final enclosure used in the prototype.

Figure 2
CAD model of the camera enclosure with IR emitter



2.3 Controlled Dark-Room Recording Protocol

All experiments used the same scene: a blue box placed on a white table in a completely dark room without external illumination. For each camera type, we recorded 20-second videos at 50, 100, 150, and 250 cm observation distances. For the software benchmarking and Kinect trials, a person walked back and forth across the field of view to provide a qualitative temporal reference for motion smoothness and perceived latency.

Event-based sensors produce events primarily in response to temporal intensity changes [18]. Therefore, for a static scene we engineered a flickering IR illumination pattern (approximately 5 ms period) to induce controlled brightness changes and enable observation.

2.4 Hardware Benchmarking Cameras

We evaluated four sensing modalities:

1) Thermal camera: FLIR T540-class device, used as a high-quality reference for visibility in complete darkness [14].

2) Xbox Kinect v1: consumer depth/IR sensor family [15]. To keep comparisons focused on intensity-based night- vision, we do not include depth outputs in the primary image-quality comparison.

3) Event-based camera: Prophesee Metavision EVK4-HD family [16], leveraging event-based sensing principles [17], [18].

4) Our low-cost NIR camera: a consumer camera con- figured for NIR sensitivity, evaluated with the same illumination source as the Kinect and software tests.

2.5 Proposed Image Enhancement Pipeline

In our hardware VR setup, we developed a real-time "Grid Mode" visualization, as can be seen in Fig. 3. This debugging tool stitches the output of every intermediate processing step into a single 2 x 3 video matrix, allowing simultaneous comparison of effects of filters implemented in our pipeline. The following filters were implemented to provide the most effective results in this exact order:

1) Baseline (Grayscale Input): We observed that the raw input suffered from extremely low dynamic range. Details in shadowed areas were indistinguishable from the background.

2) CLAHE (Local Contrast): Application of CLAHE successfully recovered details in the dark regions. However, this step introduced a significant side effect: the amplification of sensor noise, resulting in visible "grain" or "salt-and-pepper" artifacts.

3) Gaussian Denoising: By applying a 3 x 3 Gaussian blur, we successfully suppressed the grain introduced by CLAHE. While this resulted in a cleaner image, it inevitably softened object boundaries, slightly reducing the sharpness.

4) Laplacian Sharpening: To counteract the smoothing from the denoising step, Laplacian filtering was applied. This restored the structural edges of obstacles without re-introducing the background noise, effectively "popping" objects from the background.

5) Linear Gain (Brightness Boost): Finally, a linear scaling factor of 1.25 (+25%) was applied. This ensured that the final image is brighter

This distinct separation of concerns, where one filter cor- rects the artifacts introduced by the previous one-proved superior to attempting to solve all issues with a single complex algorithm.

Figure 3
Demonstration of filters' effect

2.6 Software Benchmarking Pipelines

To target lightweight embedded deployment, this manuscript intentionally avoids neural networks and other AI/ML models. We benchmarked the following classical real-time pipelines at 640×480 resolution:

1. Raw grayscale (no enhancement).
2. CLAHE-only [4].
3. Bilateral + CLAHE [4], [7].
4. NLM + CLAHE [4], [8].
5. Retinex SSR grounded in Retinex theory [9] and center/surround Retinex enhancement [10].
6. Retinex SSR + percentile normalization (RetinexSSR_Pctl), using robust intensity scaling to reduce sensitivity to outliers (contrast scaling background [1]).
7. Proposed: rotate, grayscale, CLAHE, Gaussian smoothing [23], Laplacian-based sharpening [1], [24], and Linear Gain (Brightness boost).
8. Proposed_V2: Retinex-driven enhancement integrated with lightweight post-processing to approach the objective quality of RetinexSSR_Pctl while preserving embedded feasibility [9], [10].

2.7 Runtime and Quality Metrics

We present the following metrics by which we evaluated the different image enhancement methods:

1. Runtime: average ms/frame and FPS.
2. Entropy: Shannon entropy as a compact proxy for information content [25].
3. Edge strength: mean gradient magnitude (Sobel operator implementation as commonly described in classical imaging texts [1]).
4. Sharpness proxy: variance of Laplacian, commonly used in autofocus and sharpness estimation [26].
5. RMS contrast: intensity standard deviation, consistent with contrast interpretation in complex images [27].
6. Mean intensity: used to detect near-black runs due to auto-exposure/auto-gain drift.

Each method ran for a fixed time window (N = 20 s) and excluded an initial warm-up interval to reduce bias from exposure stabilization.

3. Results

3.1 Software Benchmarking

In Table I, we report both runtime and image-quality statistics computed per frame and averaged across four scenes. Runtime is summarized by milliseconds per frame and frames per second (FPS). Visual-information content is quantified using Shannon entropy (higher indicates a richer intensity dis- tribution). Structural detail is captured by mean edge strength (average gradient magnitude) and the variance of the Laplacian (LapVar), which increases when high-frequency components and fine boundaries are preserved. Global contrast is mea- sured using RMS contrast, while mean intensity (MeanInt) is included as a guard metric to flag near-black or saturated runs caused by auto-exposure/auto-gain drift; the number of processed frames contextualizes each average.

Overall, Table I highlights a clear trade-off between en- hancement strength and real-time feasibility on the Jetson Orin NX. RawGray is fastest (1192.78 FPS) but exhibits the weak- est visibility-related metrics (entropy 2.792, edge 3.73, RMS 9.84, mean intensity 7.08), confirming that raw NIR stream does not guarantee high night-vision quality. CLAHE provides a strong baseline improvement with minimal overhead (568.13 FPS), substantially increasing entropy (4.506) and contrast (RMS 15.89), while Bilateral+CLAHE and NLM+CLAHE further target noise at the cost of speed [2], [4], [7]. Most notably, NLM+CLAHE drops to 4.27 FPS despite similar entropy/edge levels to the other CLAHE-based variants [8]. RetinexSSR produces a large structural gain (edge 31.14, LapVar 6090.6) and raises mean intensity (107.19), but its runtime (23.32 FPS) limits interactive operation [9], [10]. The percentile variant pushes objective scores even higher (entropy 7.195, edge 96.53, RMS 42.39) yet runs at only 12.63 FPS. Proposed delivers the best balance for embedded deployment: high frame rate (269.36 FPS, 3.738 ms/frame) while improving all quality metrics over CLAHE (entropy 4.658, edge 19.36, LapVar 1155.7, RMS 19.81) with a moderate mean intensity (22.06), indicating strong detail recovery without excessive global brightening. ProposedV2 achieves the highest objective enhancement overall (entropy 7.443, RMS 50.03, LapVar 73331.7) but falls to 75.34 FPS, suggesting it is best suited for scenarios where quality is prioritized over maximum frame rate.

3.2 Qualitative Evaluation

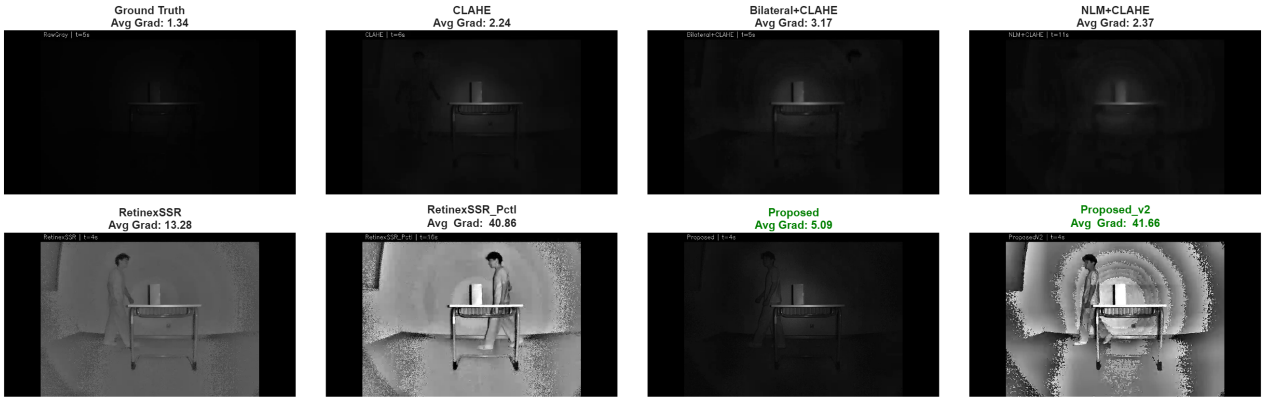
Figure 4 provides a comparison of the output frames produced by each enhancement pipeline under the same conditions. While the objective metrics in Table I quantify information content, contrast, and structural sharpness, the visual results highlight practical perceptual differences such as noise amplification, haloing/over-brightening, and the continu-

ity of scene edges. In particular, the figure illustrates how stronger enhancement methods reveal previously hidden structures, but may also introduce artifacts, whereas the proposed pipeline aims to recover detail while maintaining natural brightness and suppressing excessive noise.

Table 1
Mean software benchmark across four distances (100-250 cm) at 640× 480 on Jetson Orin NX. Lower is better for ms/frame. Higher is better for FPS, entropy, edge strength, LapVar, RMS contrast, and mean intensity

Method	ms/fra-me	FPS	En-tropy	Edge	LapVar	RMS	Mean Int	Fra-mes
RawGray	0.839	1192.78	2.792	3.73	197.2	9.84	7.08	897.2
CLAHE	1.761	568.13	4.506	10.7	581.9	15.89	17.65	877.7
Bilateral+CLAHE	11.432	87.56	4.544	5.72	208.7	16.29	17.79	618.2
NLM+CLAHE	234.141	4.27	4.333	3.93	199.3	15.21	16.7	71.75
RetinexSSR	42.928	23.32	5.703	31.14	6090.6	15.05	107.19	257.2
RetinexSSR-Pctl	79.222	12.63	7.195	96.53	65481.4	42.39	133.88	157.0
Proposed	3.738	269.36	4.658	19.36	1155.7	19.81	22.06	633.5
Proposed (V2)	13.274	75.34	7.443	89.27	73331.7	50.03	124.27	436.5

Figure 4
Visual comparison of the evaluated image enhancement pipelines



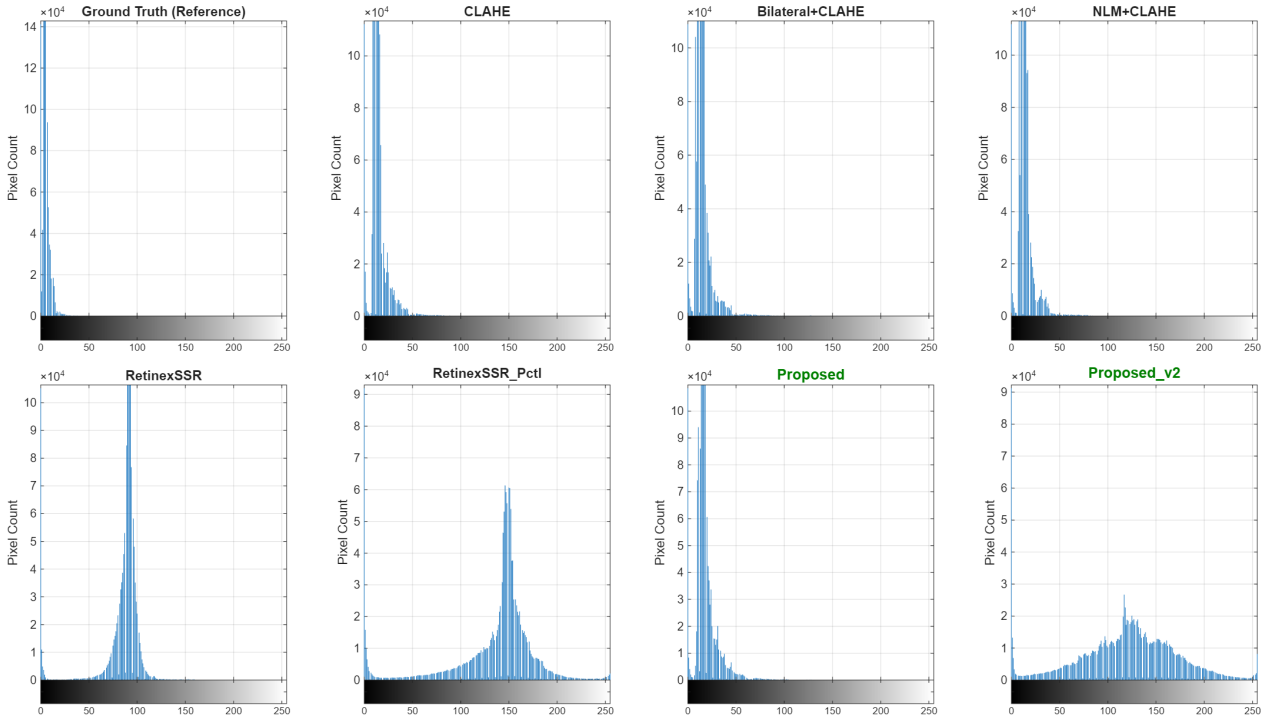
To further validate the contrast improvements objectively, histogram distributions were analyzed across the pipelines, as illustrated in Fig. 5. The raw grayscale frames consistently exhibit a narrow, concentrated pixel distribution, indicative of poor illumination and constrained dynamic range. Enhancement pipelines such as CLAHE, Retinex, and the Proposed methods actively stretch this distribution across the full 0–255 intensity spectrum. Notably, the Proposed version 2 (Proposed v2) method achieves a smoother and more evenly distributed histogram than standard CLAHE. This flattened distribution mathematically demonstrates that the proposed approach better utilizes the available dynamic range, mitigating the severe intensity spikes that typically correspond to noise amplification in under-exposed regions.

To extract the structural boundaries of the images, the widely adopted Canny edge detector was utilized [28], presented in Fig. 5. While spatial filtering techniques like Bilateral and NLM smoothing inherently reduce the edge strength metric by blurring fine textures, the RetinexSSR and Proposed algorithms significantly increase it. Visual analysis applying the Canny edge detector corroborates these mathematical findings; the edge maps extracted from the Proposed and Proposed V2 pipelines exhibit more continuous, pronounced, and well-defined structural

boundaries compared to both the raw and CLAHE-only outputs. This confirms that, despite the computational overhead required for the more complex pipelines, the proposed methodologies successfully extract hidden structural details that are otherwise lost in low-light conditions.

A practical limitation observed during repeated runs is that auto-exposure/auto-gain drift can cause near-black frames, producing misleading metrics if not detected. Mean intensity was therefore included as a guard metric to identify invalid runs and improve repeatability.

Figure 5
Comparison of histograms of different methods



3.3 Hardware Benchmarking

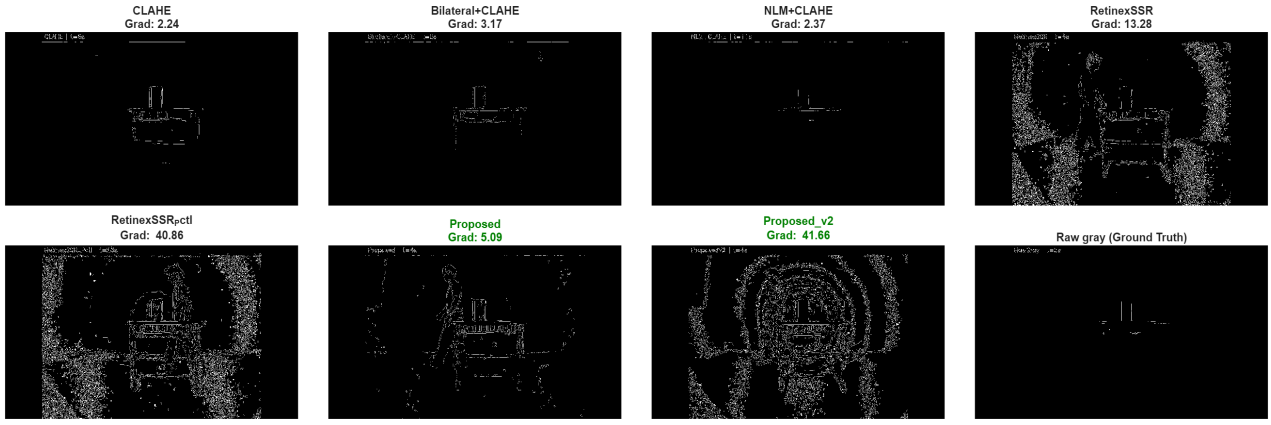
To complement software benchmarking across image enhancement pipelines, we provide a qualitative visual comparison of the raw imagery captured by the tested hardware Figure 6 compares target visibility and apparent contrast under complete darkness across sensing modalities and illumination requirements, including the effect of increasing distance from the point of observation.

The thermal camera provided the clearest visibility of the target under complete darkness and required minimal tuning, demonstrating the inherent advantage of LWIR thermal sensing for night vision [14]. However, it is substantially more expensive, heavier, and less energy efficient than a consumer NIR camera, and frame acquisition required additional integration effort.

The Kinect v1 provides IR capability but required significantly stronger illumination to achieve a comparably clear intensity image. Depth sensing can reveal structure in low-light scenarios, but we excluded depth outputs to keep comparisons focused on intensity-based night-vision imagery [15].

Event-based sensing offers extremely low latency and high dynamic range [17], [18], but static objects may produce few events under constant illumination. By engineering flickering IR illumination (5 ms period), we achieved strong visibility of static targets. Nevertheless, without temporal modulation and poorer spatial resolution, the proposed NIR frame-based cam- era remains more straightforward for static-scene observation. Overall, the proposed low-cost NIR camera and lightweight enhancement provide a practical balance among cost, deployability, energy consumption, and real-time usability

Figure 6
Comparison of edge maps of different methods



3.4 Real-Time Performance Evaluation

The system demonstrated stable operation in real time. Experiments confirmed:

1) Software processing does not result in a significant loss of frame rates.

2) The main bottleneck is data transmission over the network, not the complexity of the algorithms.

3) As the broadcast quality deteriorates, latency decreases.

4) A drop in stability was observed under Wi-Fi interference.

Therefore, the software pipeline has proven its suitability, and optimization is more likely to be required on the data transmission side than on the processing side.

3.5 Summary of Experimental Findings

Our experimental validation yields four primary conclusions regarding the feasibility of low-cost night vision:

• Spectral Sensitivity: The mechanical removal of the IR-cut filter is the foundational enabler. Without this hardware modification, the software pipeline receives insufficient photon data to recover a usable signal.

• Pipeline Synergy: The specific sequence of Grayscale → CLAHE → Denoise → Sharpen proved superior to global enhancements. Each step successfully mitigates the artifacts introduced by the previous one (e.g., Denoising correcting CLAHE-induced grain).

• Latency Bottleneck: The software pipeline, running on the Jetson platform with V4L2 optimization, introduces negligible latency. The primary bottleneck for real-time operation remains the network bandwidth required to stream the MJPEG feed to the HoloLens.

• Operational Viability: The system successfully allows for navigation in near-zero light conditions, provided that the active IR emitter is engaged to establish a minimum Signal-to-Noise Ratio for the contrast enhancement algorithms.

4. Discussion

The results of this development phase confirm that a low-cost webcam can be successfully repurposed as a night-vision imaging device when combined with a targeted infrared modification and an efficient real-time enhancement pipeline. The 850 nm IR-pass filter effectively removed visible-light interference and extended the sensor's spectral sensitivity into the near-infrared range.

The primary contribution of this work lies in the specific software pipeline developed to process raw IR-dominant frames under severe low-light conditions. In particular, the CLAHE step improved local contrast, while the denoise-then-sharpen sequence preserved edges without amplifying high-frequency noise, and the final linear gain provided a consistent brightness increase across the scene.

A key insight from our evaluation is the necessity of a "Denoise-then-Sharpen" approach. Applying sharpening directly to the noisy low-light input consistently amplified sensor noise, whereas denoising first and then restoring edges produced clearer contours and more stable visual structure, highlighting the trade-off between detail recovery and noise suppression.

Despite these successes, limitations remain. The system is heavily reliant on active infrared illumination; without a dedicated emitter, image quality degrades rapidly and usable range is reduced. In addition, overall performance is constrained by the

camera sensor's inherent noise and dynamic range, and by practical bandwidth limits when streaming at higher resolutions and frame rates.

The acquired image was projected onto the Augment Reality Headset (Figure 7).

Figure 7
Visual output of Hololens 2

5. Conclusions

This paper benchmarked low-light imaging using both sensor-level hardware and software enhancement pipelines under a controlled dark-room protocol on embedded hardware. We intentionally avoided AI/ML models to prioritize lightweight real-time operation. Results show that RetinexSSR Pctl and the Retinex-integrated ProposedV2 can yield strong objective quality metrics, but are substantially slower than the proposed CLAHE-based pipeline. The proposed CLAHE-based pipeline provides the best balance of real-time FPS and image clarity under our constraints and remains suitable for streaming and mixed-reality visualization. Future work will focus on integrating Retinex-style illumination normalization into the real-time pipeline while reduc- ing computational cost (e.g., approximations, downsampling strategies, and robust normalization) so that Retinex-level quality can be approached at higher FPS. At the system level, we will improve mixed-reality usability by rendering the enhanced video as a spatially stable element in HoloLens 2 (world-locked/anchored visualization) [22].

Acknowledgement

In collaboration with Tactile Lab and ARMS Lab.

Funding

This work was supported by the Science Committee of the Ministry of Science and Higher Education of the Republic of Kazakhstan (Grant No. AP23485994)

Conflicts of Interest

All authors have read and agreed to the published version of the manuscript

Author Contributions

Conceptualization, Z.A.; Methodology, Z.A.; Software, Z.A.; Validation, Z.A.; Formal Analysis, Z.A.; Investigation, Z.A.; Resources, I.U.; Data Curation, Z.A.; Writing – Original Draft Preparation, Z.A.; Writing – Review & Editing, I.T., A.Y., Z.R. and A.B.; Visualization, Z.A.; Supervision, Z.K.; Project Administration, Z.A.; Funding Acquisition, Z.K.

References

1. R. C. Gonzalez and R. E. Woods, Digital Image Processing, 4th ed. New York, NY, USA: Pearson, 2020.
2. P. Kamboj and V. Rani, "A brief review of image denoising techniques," Vis. Comput. Ind., Biomed., Art, vol. 2, no. 8, 2019.
3. S. M. Pizer et al., "Adaptive histogram equalization and its variations," Comput. Vis. Graph. Image Process., vol. 39, no. 3, pp. 355–368, 1987.
4. K. Zuiderveld, "Contrast limited adaptive histogram equalization," in Graphics Gems IV. San Diego, CA, USA: Academic Press, 1994, pp. 474–485.
5. Y. Li, J. Lu, J. Wang, Z. Miao, and W. Xu, "Night Vision Image Contrast Enhancement Base on Adaptive Dynamic Histogram," in Proc. 4th Int. Conf. on Digital Manufacturing & Automation (ICDMA), 2013, pp. 823– 828, doi: 10.1109/ICDMA.2013.195.
6. P. Perona and J. Malik, "Scale-space and edge detection using anisotropic diffusion," IEEE Trans. Pattern Anal. Mach. Intell., vol. 12, no. 7, pp. 629–639, 1990
7. C. Tomasi and R. Manduchi, "Bilateral filtering for gray and color images," in Proc. IEEE Int. Conf. Comput. Vis. (ICCV), 1998, pp. 839– 846.

8. A. Buades, B. Coll, and J.-M. Morel, “A non-local algorithm for image denoising,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), 2005, pp. 60–65.

9. E. H. Land and J. J. McCann, “Lightness and retinex theory,” J. Opt. Soc. Am., vol. 61, no. 1, pp. 1–11, 1971.

10. D. J. Jobson, Z. Rahman, and G. A. Woodell, “Properties and perfor- mance of a center/surround retinex,” IEEE Trans. Image Process., vol. 6, no. 3, pp. 451–462, 1997.

11. X. Guo, Y. Li, and H. Ling, “LIME: Low-light image enhancement via illumination map estimation,” IEEE Trans. Image Process., vol. 26, no. 2, pp. 982–993, 2017.

12. C. Wei, W. Wang, W. Yang, and J. Liu, “Deep retinex decomposition for low-light enhancement,” in Proc. Brit. Mach. Vis. Conf. (BMVC), 2018.

13. C. Guo, C. Li, J. Guo, C. C. Loy, J. Hou, S. Kwong, and R. Cong, “Zero- reference deep curve estimation for low-light image enhancement,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), 2020, pp. 1780–1789.

14. Teledyne FLIR, “FLIR T540 professional thermal camera,” 2026. [On- line]. Available: https://www.flir.com/products/t540/

15. Z. Zhang, “Microsoft Kinect sensor and its effect,” IEEE MultiMedia, vol. 19, no. 2, pp. 4–12, 2012.

16. Prophesee, “EVK4-HD (Metavision evaluation kit) documentation,” 2023. [Online]. Available: https://docs.prophesee.ai/stable/hw/evk/evk4.html

17. P. Lichtsteiner, C. Posch, and T. Delbr¨uck, “A 128×128 120 dB 15 μs latency asynchronous temporal contrast vision sensor,” IEEE J. Solid- State Circuits, vol. 43, no. 2, pp. 566–576, 2008.

18. G. Gallego et al., “Event-based vision: A survey,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 44, no. 1, pp. 154–180, 2022.

19. “Low-Cost IR Night Vision System – Source Code,” GitHub repository, 2025. [Online]. Available: https://github.com/YevgeniyDev/lowcost_nightvision_camera.git

20. NVIDIA, “Jetson Orin NX series data sheet,” 2022. [Online]. Available: https://developer.nvidia.com/downloads/ jetson-orin-nx-series-data-sheet

21. G. Bradski, “The OpenCV library,” Dr. Dobb’s Journal of Software Tools, 2000.

22. Microsoft, “HoloLens 2 hardware,” 2023. [Online]. Available: https://learn.microsoft.com/en-us/hololens/ hololens2-hardware

23. A. P. Witkin, “Scale-space filtering,” in Proc. Int. Joint Conf. Artif. Intell. (IJCAI), 1983, pp. 1019–1022.

24. D. Marr and E. Hildreth, “Theory of edge detection,” Proc. Royal Soc. London B, vol. 207, no. 1167, pp. 187–217, 1980.

25. C. E. Shannon, “A mathematical theory of communication,” Bell Syst. Tech. J., vol. 27, pp. 379–423 and 623–656, 1948.

26. J. L. Pech-Pacheco, G. Crist´obal, J. Chamorro-Mart´ınez, and J. Fern´andez-Valdivia, “Diatom autofocusing in brightfield microscopy: a comparative study,” in Proc. Int. Conf. Pattern Recognit. (ICPR), 2000, pp. 3318–3321.

27. E. Peli, “Contrast in complex images,” J. Opt. Soc. Am. A, vol. 7, no. 10, pp. 2032–2040, 1990.

28. J. Canny, “A computational approach to edge detection,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 8, no. 6, pp. 679–698, 1986.

Information about the authors:

Zaki Al-Farabi – is an undergraduate student at Nazarbayev University (ORCID iD: 0009-0001-6300-4119).

Ilyas Umurbekov – holds a BS degree in Robotics from Nazarbayev University (ORCID iD: 0009-0003-1147-3196).

Iliyas Tursynbek – holds a PhD degree from Nazarbayev University (ORCID iD: 0000-0003-3753-9969).

Adilet Yesbayev – PhD, is a research assistant at Tactile Robotics Lab (ORCID iD: 0000-0002-7572-034X).

Zhanibek Rysbek – PhD, is a research assistant at Tactile Robotics Lab (ORCID iD: 0009-0003-2169-5139).

Almaskhan Baimyshev – PhD, is a research assistant at Tactile Robotics Lab (ORCID iD: 0000-0002-8494-3676).

Zhanat Kappassov – is an Associate Professor at Nazarbayev University and has over 15 years of experience in robotics, tactile sensing, artificial intelligence, and machine learning. His research interests include robotics, computer vision, and deep learning applications. He is a senior member of the Institute of Electrical and Electronics Engineers (IEEE). Dr. Kappassov has received numerous awards, including the Best PhD Thesis Award in France in 2017 and the Kazakhstani National Award for young scientists, the Kunaev Award, in 2025 (e-mail: zhkappassov@nu.edu.kz. ORCID iD: 0000-0003-3262-3993).

Submission received: 14 January, 2026
Revised: 16 March, 2026
Accepted: 16 March, 2026

IRSTI 20.15.05 https://doi.org/10.26577/jpcsit4220264

Zhansaya Segizbayeva[1*], Marek Miłosz[2]

[1]Al-Farabi Kazakh National University, Almaty, Kazakhstan
[2]Lublin University of Technology, Lublin, Poland
*e-mail: segizbayeva.zhansaya@gmail.com

COMPUTATIONAL MODEL OF KAZAKH VOWEL–CONSONANT HARMONY

Abstract. This study presents a computational model of morphological generation based on the agglutinative nature of the Kazakh language and vowel–consonant harmony. In this work, we model the Kazakh vowel–consonant harmony system, which governs allomorphic selection of suffixes based on both vowel and consonant properties. Using a rule based approach, the model generates verb and noun forms across various grammatical categories (tense, aspect, case, possession). Experimental results demonstrate the model's high efficiency and full compliance of the generated word forms with vowel harmony rules. Specifically, thousands of combinations were created for derivative words (DTL), possession (POSS1, POSS2), and case (CASE) categories. This research makes a significant contribution to the development of natural language processing (NLP) morphological analysis and automatic translation systems for the Kazakh language.

Keywords: kazakh language, computational linguistics, morphological generation, vowel–consonant harmony, harmony law, agglutinative languages, and natural language processing.

1. Introduction

The Kazakh language is an agglutinative language belonging to a diverse language family, characterized by a rich morphological structure and a complex sound system. The main sound system is the law of harmony, ensuring vowel harmony within a single syllable. This rule underlies word formation, controlling the correct connection of suffixes to the root [1], [2]. The rapid development of digital technologies has increased the importance of studying Kazakh natural language processing (NLP) performance. However, these features of the Kazakh language do not facilitate the ready use of natural language processing models, that is, special computers are needed to understand the language's structure [3], [4]. Computational modeling of syllable and sound harmony is key for developing practical tools such as morphological analysis, filters, automatic translations, and text editors. Early studies relied on rule-based approaches, including finite state automata and two-level morphology [1]. While these models describe morphological regularities, creating them requires much work. Recent data indicate that neural systems and statistical methods in hybrid models are well-developed. Such models can effectively handle deviations from the language norm, but require a large database to achieve high-quality results. Still, creating generative models that preserve the regularities of syllable andsound harmony in Kazakh is an urgent problem.

The study aims to develop a computational model for Kazakh morphological rules and regularities of syllable and sound harmony and experimentally assess its effectiveness.

The study's objectives are: 1) Develop an algorithm for forming Kazakh word class forms (nouns, verbs, pronouns, prepositions, moods). 2) Create rules considering syllable types, sound harmony, and inflectional differences, including solid-soft and labial variations. 3) Conduct generative-numerical analysis for various grammatical categories. 4) Identify the model's strengths and weaknesses based on results.

Vowel–consonant harmony in the Kazakh language is a complex phonological process that regulates the harmony of vowels, and sometimes consonants, in a word. This is a feature characteristic of Turkic languages, where all vowels in a word usually harmonize with the first vowel of the root. Such harmony ensures the phonetic integrity of the language and plays an important role in the agglutinative structure (i.e., the system of conveying grammatical meaning through affixes). McCollum (2018) examined the gradual and local nature of consonant harmony in Turkic languages and roposed a variance analysis utilizing the "Maximum Entropy Harmonic Grammar" [5].

There are two main types of harmony patterns in the Kazakh language: palatal harmony (solid-soft harmony) and labial harmony (lip harmony). According to the law of palatal harmony, all vowels in a word must be solid (/'a', 'o', ' ū', 'y', 'u'/) or thin (/ä', ' ö', ' ü', 'e', 'i'/). There is also lip sync, but it is considered secondary and regulates the correspondence of the lip features of vowels.

Zhusupov and Saparova (2024) divide syllable and sound sync in Turkic languages into four types: singarmolinguohard, singarmolinguosoft, singarmolinguolabiohard, and singarmolinguolabiosoft, which indicates that it is a universal prosodic feature [6]. They conclude that the phonostylistics of the Kazakh language is mainly singarmolinguo, and labial syncharmonism is weak.

They compare this to non-syntactic languages, such as Russian, and show that, in addition to vowels, consonant harmony also plays an important role in the morphophonemic structure of the Kazakh language. This includes phenomena such as progressive and regressive assimilation, in which consonants adapt to neighboring sounds and adopt their articulatory properties. Such consonant consonance patterns impose strict morphophonemic constraints on the combination of roots and suffixes in addition to vowel consonance [7].

The choice of a particular affix form is often determined by these phonological rules, i.e., it corresponds to the sounds preceding the affixes to which it is attached. If the root contains thick and thin vowels, then the vowel in the final syllable determines the type of consonant. These patterns are often irregular and do not fully obey the law of consonance.

The complexity of the Kazakh language's syllabic and phonetic harmony—its agglutinative nature, the coexistence of several types of harmony, and the processes of mutual assimilation of vowels and consonants—poses significant challenges for computational linguistics. Therefore, any effective computational model for the Kazakh language must not only accurately reflect these complex phonological patterns, but also take into account the difficulties arising from the writing system.

The first and most fundamental approaches to computationally modeling vowel–consonant harmony in Kazakh are based on rule-based systems. Boundary state modifiers (BSTs) and two-level morphology are often used. Such models are especially useful for agglutinative languages such as Kazakh, since morphological processes can be systematically and accurately described by boundary rules and state transitions.

Tukeyev (2015) used finite-state automata-based models to formally describe the morphological structure of the Kazakh language [8]. The study examined the problems of morphological analysis of words and checking the completeness of conjugations using an automatic approach, and proposed effective ways to formalize morphological processes characteristic of agglutinative languages.

The work of Kessikbayeva and Chichekli (2014) [9] is based on two-level morphology implemented using Xerox boundary state tools. This analyzer describes in detail the patterns of suffix change and suggests rules for using different suffix forms according to compatibility. Similarly, Kairaqbay and Zaurbekov (2013) proposed a boundary state-based approach to analyzing the noun paradigm in Kazakh [10]. Their model takes into account the morphophonemic constraints (vowel-consonant and voiceless patterns) specific to the Kazakh language.

Due to the limitations of rule-based systems, data-driven and hybrid approaches are currently being actively developed. These models combine the accuracy of rule-based methods with the flexibility offered by statistical and machine learning techniques.

Zhetkenbai et al. (2025) proposed a hybrid computational model for formatting and predicting the morphological inflection of Kazakh nouns based on the new Latin alphabet [11]. This model integrates rule-based grammatical formatting with a Bayesian regularized backpropagation neural network (BR-BPNN). The BR-BPNN model showed high accuracy (91.5%) and specificity (89.4%).

Stoyer et al. (2023) conducted a study to describe vowel harmony from an information theory perspective [12]. They suggested an approach based on neural recurrent language models (LSTMs).

Overall, the literature emphasizes the critical role of rule-based approaches (FST, two-level morphology) in accurately modeling synchronization for Kazakh morphological production, alongside the growing importance of hybrid and neural models (BR-BPNN, Transformer, LLM) in enhancing flexibility and accuracy. The current work continues this tradition by presenting a precise computational model for syllable and sound harmony generation.

2. Materials and methods

The proposed computational model uses a rule-based approach to implement the morphological formation of the Kazakh language. This approach is based on the regularity of morphological processes in agglutinative languages and formally implements

the generative aspect of the concept of finite-state transducers (FST) [1] [3]. The primary objective of the model is to generate the correct word form for a given root and set of grammatical categories, strictly adhering to the laws of syllable and sound harmony.

The architecture of the model is based on the interaction of the following main components:

- Phonetic-lexical resources: Phonetic groups based on the classification of vowels and consonants and the intralinguistic features of the root, for example, the thickness/thickness of the root.

- Morphotactic rules: A set of rules that determine the order and permissible combinations of roots and suffixes.

- Syllable and sound harmony control logic: Conditional logical operators that control the choice of a solid/soft or labial/non-labial version of a suffix. This logic depends on the characteristics of the vowel in the final syllable of the root.

This structure allows for computationally accurate and efficient modeling of the morphophonemic processes of the Kazakh language.

Below are the subsets of letters that play an important role in the rules of vowel harmony of words with nominal roots [13].

vsolid = [‘a’, ‘o’, ‘ ū’, ‘y’, ‘u’]; vsoft = [‘ ä’, ‘ ö’, ‘ ü’, ‘ı’, ‘e’, ‘u’, ‘i’]; cdeaf = [‘p’, ‘k’, ‘q’, ‘t’, ‘s’, ‘ ş’, ‘h’]; cvoiced = [‘b’, ‘v’, ‘ g’, ‘ ğ’, ‘d’, ‘j’, ‘z’]; cv1 = [‘b’, ‘v’, ‘g’, ‘ ğ’, ‘d’]; cv2 = [‘j’, ‘z’]; cvoiced1 = [‘b’, ‘v’, ‘g’, ‘d’]; csonor = [‘l’, ‘m’, ‘n’, ‘ ñ’, ‘r’, ‘ı’, ‘u’]; cs1 = [‘m’, ‘n’, ‘ñ’]; cs2 = [‘l’, ‘r’, ‘ı’, ‘u’]; cs3 = [‘l’, ‘m’, ‘n’, ‘ñ’]; cs4 = [‘r’, ‘ı’, ‘u’].

Table 1
Rules for forming words with plural endings

LAR/LER rules	TAR/TER rules

if stem[-1] in (vsolid or vsoft or cs4):
if stem[-1] in vsolid and ending[1] in vsolid:
if ending[:3] == ‘lar’:
endingtl = ending
#print(‘11.a solid’)
if stem[-1] in vsoft and ending[1] in vsoft:
if ending[:3] == ‘ler’:
endingtl = ending
	if stem[-1] in (cdeaf or cv1):
	if stem[-2] in vsolid and ending[1] in vsolid:
	if ending[:3] == ‘tar’:
	endingtl = ending
	#print(‘11.a solid’)
	if stem[-2] in vsoft and ending[1] in vsoft:
	if ending[:3] == ‘ter’:
	endingtl = ending

Table 1. A brief explanation of the rules for the plural endings (-lar/–ler, –tar/–ter). This table provides phonetic rules that automate the correct selection of plural endings. The rules are based on the final sound of the word and the vowel harmony [16].

– LAR / –LER

If the word ends in a vowel or a voiced/voiced consonant: The last sound is thick → lar.

The last sound is thin → ler. The code checks the consonance of the ending using ending[:3].

– TAR / –TER

If the word ends in a hard consonant: The previous sound is thick → tar. The previous sound is thin → ter. This rule is fixed by the conditions ending[:3] == ‘tar’/’ter’. The algorithm automatically selects the correct form of the plural ending, taking into account the law of consonance in the Kazakh language.

These 2 tables provide consonant rules that determine the correct version of the possessive conjunction when the final sound of the word is a vowel. The algorithm makes a choice taking into account the thickness or thinness of the final and preceding sounds of the root, as well as the presence of the last letter “u”. The algorithm checks the correspondence of the possessive conjunction (1st person, 2 nd person singular/plural) to the thickness or thinness of the consonant and the final sound of the root, and automatically selects the correct conjunction.

This rule automatically generates classified forms of the present tense depending on whether the vowel in the final syllable of the verb stem is thick or thin.

Table 2
Rules for forming words with possessive endings

Rules for possessive endings (the last letter is a vowel)

if stem[-1] in vsolid and ending[1] in vsolid and stem[-2] in vsolid and stem[-1] == ‘u’:
if ending == ‘m’ or ending == ‘ń’ or ending[:3] in (‘ńyz’, ‘myz’) or (ending[:2] == ‘sy’):
endingtl = ending
if stem[-1] in vsolid and ending[1] in vsolid:
if ending == ‘m’ or ending == ‘ń’ or ending[:3] in (‘ńyz’, ‘myz’) or (ending[:2] == ‘sy’):
endingtl = ending
if (stem[-1] in vsoft and ending[1] in vsoft and stem[-2] in vsoft and stem[-1] == ‘u’):
if ending == ‘m’ or ending == ‘ń’ or ending[:3] in (‘ńiz’, ‘miz’) or (ending[:2] == ‘si’):
endingtl = ending
if (stem[-1] in vsoft and ending[1] in vsoft and stem[-2] in vsoft and stem[-1] == ‘u’):
if ending == ‘m’ or ending == ‘ń’ or ending[:3] in (‘ńiz’, ‘miz’) or (ending[:2] == ‘si’):
endingtl = ending

Table 3
Verb (classified forms). Rule of the present tense (Present Simple)

Verb (classified forms). Rules for the present tense (Present Simple)
if stem[-2] in vsolid: for ending in pres_spl_sg_vsolid: forms.append(stem + ending) for ending in pres_spl_pl_vsolid: forms.append(stem + ending) if stem[-2] in vsoft: for ending in pres_spl_sg_vsoft: forms.append(stem + ending) for ending in pres_spl_pl_vsoft: forms.append(stem + ending)

If the previous sound of the root is solid (vsolid):

– The list of suffixes for solid persons (pres_spl_sg_vsolid and pres_spl_pl_vsolid) is added, and all classified forms are generated.

If the previous sound is soft (vsoft):

– The list of suffixes for soft persons (pres_spl_sg_vsoft and pres_spl_pl_vsoft) is removed, and all forms are generated.

This rule automatically selects the pronoun forms –ğan/–gen/–qan/–ken, as well as the forms –atyn/–etyn, –ytyn/–ytyn, based on the final sounds of the root.

Table 4
Participle (PP) rules

Participle (PP). -ĞAN/-GEN/-QAN/-KEN, -ATYN/-ETIN, -ITYN/-ITIN rules

if stem[-2] in vsolid and stem[-1] in csonor and ending[:3] == ‘gan’:
endingtl = ending
if stem[-2] in vsoft and stem[-1] in csonor and ending[:3] == ‘gen’:
endingtl = ending
if stem[-2] in vsolid and stem[-1] in cdeaf and ending[:3] == ‘qan’:
endingtl = ending
if stem[-2] in vsoft and stem[-1] in cdeaf and ending[:3] == ‘ken’:
endingtl = ending
if stem[-2] in vsolid and stem[-1] in csonor and ending[:4] == ‘atyn’:
endingtl = ending
if stem[-2] in vsoft and stem[-1] in csonor and ending[:4] == ‘etin’:
endingtl = ending
if stem[-1] in vsolid and ending[:4] == ‘ıtyn’:
endingtl = ending
if stem[-1] in vsoft and ending[:4] == ‘ıtın’:
endingtl = ending

–GAN / –GEN / –QAN / –KEN

Previous sound thick + last sound soft → –ğan

Previous sound thin + last sound soft → –gen

Previous sound thick + last sound hard → –qan

Previous sound thin + last sound hard → –ken

–ATYN / –ETYN

Last sound soft, root thick → –atyn

Last sound soft, root thin → –etyn

–ITYN / –ITYN

Last sound thick → –ıtyn

Last sound thin → –ıtyn

This rule ensures the automatic correct selection of the main adverb types (–a/–e/–ı, –yp/–ıp/–p, –qaly/–ğaly/–kelı/–gelı) based on the properties of the final sounds of the root.

Table 5
Verbal Adverb (VA) rules

(Verbal Adverb, VA). -A/-E/-I, -YP/-IP/-P, -QALY/-ĞALY/-KELI/-GELI
if (stem[-2] in vsolid or stem[-3] in vsolid) and stem[-1] in cdeaf \
and ending[:1] == ‘a’:
endingtl = ending
if stem[-2] in vsoft and stem[-1] in csonor and ending[:1] == ‘e’:
endingtl = ending
if stem[-1] in vsoft and ending[:1] == ‘ı’:
endingtl = ending
if (stem[-2] in vsolid or stem[-3] in vsolid) and stem[-1] in cdeaf \
and ending[:2] == ‘yp’:
endingtl = ending
if stem[-2] in vsoft and stem[-1] in csonor and ending[:2] == ‘ıp’:
endingtl = ending
if stem[-1] in vsoft and ending[:1] == ‘p’:
endingtl = ending
if (stem[-2] in vsolid or stem[-3] in vsolid) and stem[-1] in cdeaf \
and ending[:4] == ‘qaly’:
endingtl = ending
if stem[-2] in vsolid and stem[-1] in cvoiced and ending[:4] == ‘galy’:
endingtl = ending
if stem[-2] in vsoft and stem[-1] in cdeaf and ending[:4] == ‘kelı’:
endingtl = ending
if stem[-2] in vsoft and stem[-1] in csonor and ending[:4] == ‘geli’:
endingtl = ending

–A / –E / –I

Root thick + final sound hard → –a

Root thin + final sound soft → –e

Final sound soft → –ı

–YP / –IP / –P

Thick consonant + hard consonant → –yp

Thick consonant + soft consonant → –ıp

Final sound soft → –p

–QALY / –ĞALY / –KELI / –GELI

Thick + hard → –qaly

Thick + soft → –ğaly

Thick + hard → –kelı

Thick + soft → –gelı

The rule analyzes the thick-thin, hard-soft, soft properties of the root and automatically selects the correct ending for the preposition.

These rules provide automatic selection of the correct form of the imperative and conditional mood endings depending on whether the vowel in the final syllable of the verb stem is thick or soft.

Table 6
Rules of harmony of the imperative and conditional moods

Rules of harmony between the imperative and the conditional mood

if stem[-2] in vsolid and ending == ‘syn’:
endingtl = ending
if stem[-2] in vsoft and ending == ‘sın’:
endingtl = ending
if stem[-2] in vsolid and ending == ‘şy’:
endingtl = ending
if stem[-2] in vsoft and ending == ‘şı’:
endingtl = ending
if stem[-2] in vsolid and ending == ‘sa’:
endingtl = ending
if stem[-2] in vsoft and ending == ‘se’:
endingtl = ending

Imperative case **(-syn / -sın)**
If the root is thick → **–syn**
If the root is thin → **–sіn**
Imperative case -şy / -şy form
Root thick → –şy
Root thin → –şy

Conditional case **(-sa / -se)**
Root thick → **–sa**
Root thin → **–se**

Similar rules for calculating vowel harmony have been compiled for other parts of speech in the Kazakh language.

3. Results

To assess the derivational potential of the proposed computational model, a quantitative morphological analysis of word forms derived from the same root across the main word classes (noun, verb, preposition, adverb, and pronoun) was conducted to demonstrate the agglutinative-polysynthetic nature of the Kazakh language. This analysis allows us to determine the linguistic metrics of derivational productivity.

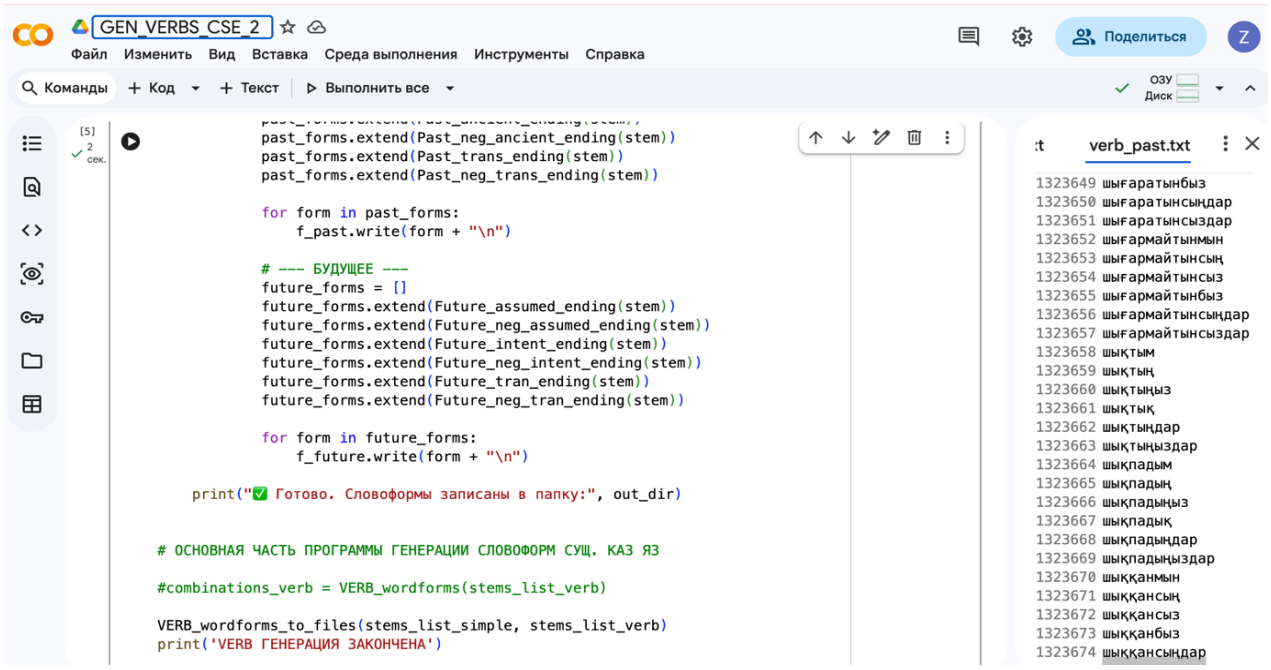
Table 7 shows the quantitative distribution of word forms across the morphological paradigms of the main word classes. All variations of syllable and sound harmony were taken into account in the derivational process.

Table 7
Quantitative indicators of word forms generated from the root word

№	Word classes	(stem)	Type of connections used	Number of generated forms
1	Noun	108	plurality (larp/ler, tar/ter), dependency (-m, -ń, -sy/-sı), accusative (-nyń, -dy, -ğa, -men ...)	21709
2	Verb	108	tense forms (present, past, future), negative, participles	2 307667
3	Voice Verbal Adverb, VA Moods	100	self, otherness, involuntary, common -a/-e/-ı, -yp/-ıp/-p, -ğaly/-kelı	844
4	Participle, PP	100	-ğan/-gen/-qan/-ken, -atyn/-etın, -uşy/-uşı, -ar/-er	17141
5	Word classes	416	-	2 347361

The results of morphological generation for six word classes were calculated using a model based on the given rules, and the results were obtained. Using the rules for each word class, all possible word forms were calculated fully automatically.

Figure 1
Number of verb endings (past tense)



As shown in Figure 1, the results obtained were 21,709 forms for nouns, 2,307,667 forms for verbs, 17,141 forms for pronouns, and 844 forms for prepositions and adverbs. A total of 2,347,3611 forms were obtained for the given word classes. This quantitatively proves the agglutinative nature of the Kazakh language and demonstrates the model's generation ability.

4. Discussion

The proposed rule-based computational model has shown high potential for fully preserving the morphophonemic processes and vowel–consonant harmony patterns of the Kazakh language in a generative paradigm. As described in the methodological formalization, the precise classification of phonetic elements and the conditional logical rules based on this classification allow for high-accuracy computational modeling of palatal vowel–consonant harmony and regressive assimilation of consonants. This approach proves the effectiveness of directly translating linguistic rules into computational algorithms [14].

Advantages of the model:

1. Formal verification and interpretation: The main advantage of the rule-based approach is its high accuracy and interpretability. Each morphological generation process can be fully verified by applying linguistic rules. This property is very important for empirical verification of theoretical conclusions in linguistic research.

2. Data independence efficiency: The fact that the model does not require a large annotated data corpus to work makes it the most effective initial approach for rapidly developing NLP tools for low-resource languages such as Kazakh.

3. Morphological space representation: As shown in the results, it is proven that the model can fully and systematically represent the agglutinative-polysynthetic morphological space of the Kazakh language by generating 2,307,667 verb forms from a lexically given root.

As mentioned in the literature review, approaches such as rule-based FST models [1], [3] and hybrid BR-BPNN [2] are used to model vowel–consonant harmony in the Kazakh language. The proposed model is similar to FST, but is focused on generation and formalizes rules using simple conditional logic.

Comparative analysis with other computational paradigms:

- With finite automaton transformers (FST): The proposed model, while having the generative capabilities of FST, implements its rules using simple conditional logic operators. This approach reduces

the complexity of building an FST and increases the speed of model development.

- With neural and hybrid models: Neural networks (Transformer, BR-BPNN) show flexibility in handlin large data-driven fluctuations and language variations [2], [15]. However, the main theoretical limitation of our model is that it cannot fully capture deviations from the law of voicelessness and secondary, sometimes unstable, manifestations of vowel–consonant harmony in imported words (e.g., "computer", "institute"). To address these issues, future research should move to a hybrid architecture by incorporating lexical features (e.g., word origins) and statistical checks into the model.

The scientific significance of this study lies in the deep formalization of the morphological syllable of the Kazakh language from a computational perspective. Quantitative analysis of syllable productivity objectively proves the morphological richness of the Kazakh language and can serve as a basis for future NLP research [15], [16].

Practical applications:

- Morphological synthesizers: The model can be used to generate correct word forms in speech systems and machine translation systems at the syllabic stage.

- Educational tools: This will be the basis for creating interactive grammar simulators for Kazakh language learners.

- Spell check tools: A large corpus of syllabic forms will help create dictionaries for checking the correct spelling of words.

5. Conclusion

This study presents a rule-based paradigm for the computational formalization of the morphological syllable of the Kazakh language. The proposed method systematically characterizes the agglutinative nature of the Kazakh language, including vowel harmony, consonant assimilation, and the multilayered structure of word formation and word inflection.

The main scientific novelty of the research is the development of a computational model capable of automatically generating hundreds of new word forms with syntactic and semantic meanings from a single lexical root in accordance with morphological laws. The model fully covers the main morphological categories, such as plural, dependent, verb, pronoun, preposition, imperative, conditional, and present tense forms.

The experimental results show that it is possible to generate morphological paradigms of nouns and verbs in a complete and error-free manner. This not only quantitatively proves the morphological richness of the Kazakh language but also opens up new opportunities for automating linguistic resources.

Thus, the morphological generation model, which combines the structural features of the Kazakh language with computational methods, lays the theoretical foundation for creating intelligent linguistic systems for agglutinative languages and expands the possibilities of integration with machine learning and neural models in the future.

Acknowledgements

The authors would like to acknowledge Ualsher Tukeyev for his valuable academic discussions, methodological feedback, and expert guidance provided during the preparation and revision of this manuscript.

Funding

This research was supported by the Ministry of Science and Higher Education of the Republic of Kazakhstan within the framework of the scientific project "Study of neural models for the formation of speech transcripts and minutes of meetings in Turkic languages" (Grant No. AP23487816).

Conflicts of Interest

The authors declare no conflict of interest.

Author Contributions

Conceptualization, M.M. and Z.S.; Methodology, M.M.; Software, M.M. and Z.S.; Validation, M.M. and Z.S.; Formal Analysis, M.M.; Investigation, M.M. and Z.S.; Resources, Z.S.; Data Curation, Z.S.; Writing – Original Draft Preparation, M.M. and Z.S.; Writing – Review & Editing, M.M.; Visualization, Z.S.

References

1. O. Özchelik, Phonetics, Phonology, Phonotactics, Morphophonology, Prosody. Published in the Encyclopedia of Turkic Languages. Brill.

2. G. B.Shoibekova, S. A.Odanova, B. M.Sultanova, & T. N. Ermekova, "Vowel harmony is the main phonetic law characteristic of Turkic languages." *International Journal of Environmental & Science Education*, 2016, vol. 11, no. 11, pp. 4617–4630.

3. U. Tukeyev, "A new computational model for Turkic languages' morphology and processing." *Journal of Problem in Computer Science and Information Technologies*, vol 1, no 1, pp. 47-54, 2023. https://doi.org/10.26577/JPCSIT.2023.v1.i1.07

4. Ya. Karabaliev, K. Kolesnikova, "Speech recognition methods in the Kazakh language: analysis of errors and possibilities of improvement." *Scientific journal of Astana IT University*, vol. 20, pp. 62–74, 2024. doi.org/10.37943/20dzgh8448

5. A. McCollum, "Vowel dispersion and Kazakh labial harmony." *Phonology*, vol. 35, no 2, pp. 239-276, 2018. doi.org/10.1017/S0952675718000052

6. M. Dzhusupov, N. Saparova, "Comparative study of synharmonic and non-syndharmonic phonostylistics (on the material of Kazakh and Russian languages)." *Journal of Linguistics, Semiotics and Semantics of the Russian State University of Technology,* vol 15, no 3, pp. 895–913, 2024. doi.org/10.22363/2313-2299-2024-15-3-895-913

7. J. Dees, "Heterogeneity in the phonologization period: data from the Kazakh language." *Proceedings of the Workshop on Turkic and Languages in Contact with Turkic,* vol. 8, pp. 74–88, 2023. doi.org/10.3765/6wbxsg62

8. U. Tukeyev, Automaton models of the morphology analysis and the completeness of the endings of the Kazakh language, Proc. TurkLang-2015, Kazan, Russia. Sep. 17-19, 2015, pp. 91–100.

9. G. Kessikbayeva, I. Cicekli, "Rule based morphological analyzer of Kazakh language." *InProceedings of the 2014 Joint Meeting of SIGMORPHON and SIGFSM*, 2014, pp. 46-54.

10. B. Kairakbay, and D. Zaurbekov. "Fsmnlp 2013-proceedings of the 11th international conference on finite state methods and natural language processing." 2013.

11. *L. Zhetkenbay, A. Sharipbay, B. Razakhova, G. Bekmanova, A. Barlybayev, A. Nazyrova, B. Yergesh,* "Formalization of Morphological Rules for Kazakh Nouns in the New Latin Alphabet," *Journal of Applied Data Sciences*, 2025. vol. 6, no 3, pp. 1999-2019.

12. J. Steuer, J. List, B. Abdullah, D. Klakow, "An Information-Theoretic Characterization of Vowel Harmony." *Transactions of the Association for Computational Linguistics*. pp. 96–109, 2023. doi:10.18653/v1/2023.sigtyp-1.10

13. U. Tukeyev. "Computational models of morphology and vowel harmony of the Kazakh language and their use for neural models." M A T E R I A L S of The International Scientific and Theoretical Conference ARTIFICIAL INTELLIGENCE IN THE SPACE OF CONTEMPORARY ART: PROBLEMS AND PROSPECTS, 11.04.2025. Baku, 2025, pp. 12-19. DOI:10.25045/ASU-CAAI.2025.01

14. 14U.Tukeyev, A. Karibayeva, Zh. Zhumanov, (2020). "Morphological Segmentation Method for Turkic Language Neural-Machine Translation." *Cogent Engineering,* vol. 7, 2020, doi.org/10.1080/23311916.2020.1856500

15. A. Aytim, "Development of methods for automatic processing systems of the Kazakh language." *KazATK Khabarshsy*, 2024, *vol. 133, no 4, pp.* 254–265, doi.org/10.52167/1609-1817-2024-133-4-254-265

16. S. Altynbek, "The need to reform the Kazakh writing system." *Educational Science,* vol. 1, no 1, pp. 1–5, 2024. doi:10.61927/igmin148

Information about the authors:

Zhansaya Eldibayevna Segizbayeva – Doctoral Candidate, Al-Farabi Kazakh National University (Almaty, Kazakhstan, e-mail: segizbayeva.zhansaya@gmail.com. ORCID: https://orcid.org/0009-0002-1324-9086).

Marek Milosz – Doctor of Philosophy (English), Professor; Lublin University of Technology (Lublin, Poland, m.milosz@pollub.pl. ORCID: https://orcid.org/0000-0002-5898-815X).

Submission received: 15 January, 2026
Revised: 11 March, 2026
Accepted: 19 March, 2026

IRSTI 28.23.37 https://doi.org/10.26577/jpcsit4220265

Yesset Zhussupov[1], Ainur Zhumadillayeva[1*], Shona Shinassylov[2], Ainur Amirtayeva[2]

[1]L.N. Gumilyov Eurasian national university, Astana, Kazakhstan
[2]Al-Farabi Kazakh National University, Almaty, Kazakhstan
*e-mail: zhumadillayeva_ak@enu.kz

COMPARATIVE ANALYSIS OF SEQUENCE-BASED AND GRAPH-BASED DEEP LEARNING FOR ANOMALY DETECTION IN MICROSERVICE TRACES

Abstract. The transition from monolithic to microservice architectures has significantly increased the complexity of system observability. Distributed tracing has emerged as an indispensable instrument for monitoring inter-service interactions, yet traditional anomaly detection methods often fail to capture the complex structural relationships inherent in these systems. This paper presents a comparative study of sequence-based approaches, specifically Long Short-Term Memory (LSTM) networks, and graph-based deep learning methods such as Graph Neural Networks (GNNs) for anomaly detection in microservice traces. The comparison combines a structured analysis of existing literature with original experimental evaluation, in which three representative models – DeepLog (LSTM), DeepTraLog (GGNN), and TraceVAE (Graph VAE) – are trained and evaluated on the TrainTicket benchmark comprising 106,312 traces from 35 microservices. We evaluate these paradigms across three dimensions: structural awareness, detection accuracy, and computational efficiency. Our experimental results, together with findings reported in prior work, confirm that graph-based models achieve higher detection accuracy (F1 up to 0.896) compared to sequence-based alternatives (F1 = 0.758), while also revealing practical trade-offs in training cost, memory consumption, and deployment complexity.

Keywords: microservice architecture, anomaly detection, graph neural networks, LSTM, distributed tracing, deep learning, observability.

1. Introduction

Modern industrial microservice systems often comprise thousands of services running across distributed containers and virtual machines [1], [2]. In contrast to conventional monolithic designs, where application logic is encapsulated in a single binary, microservices rely on loosely coupled, autonomous units communicating via lightweight protocols such as Remote Procedure Calls (RPC) or Representational State Transfer (REST) APIs [3], [4]. This architectural paradigm shift provides substantial benefits in terms of scalability, deployment flexibility, and team autonomy. However, it simultaneously introduces substantial "observability challenges," where a failure in one service can propagate through complex invocation chains, causing cascading effects across the entire distributed system [5], [6].

To mitigate these operational risks, distributed tracing technologies such as OpenTracing, Jaeger, and Apache SkyWalking have been developed to record the end-to-end execution paths of user requests as a series of hierarchically organized "spans" [7], [8]. Each span represents a unit of work within the system, containing metadata about timing, service identity, and parent-child relationships. However, the telemetry data generated by these systems is often high-dimensional, semi-structured, and characterized by complex non-linear dependencies, rendering manual troubleshooting and simple rule-based detection approaches insufficient for modern production environments [9], [10].

The emergence of machine learning (ML) and deep learning (DL) techniques has offered promising solutions for automated anomaly detection in distributed systems. Among these approaches, two paradigms have attracted considerable scholarly interest: sequence-based models, particularly Long Short-Term Memory (LSTM) networks, and graph-based models, including various Graph Neural Network (GNN) architectures. This paper offers an exhaustive evaluation of these two paradigms, examining their relative strengths and limitations across multiple evaluation dimensions.

The contribution of this paper is twofold. First, we provide a structured analytical comparison of

sequence-based and graph-based paradigms, synthesizing detection accuracy and computational efficiency findings reported across multiple independent studies. Second, we complement this literature-based analysis with original experimental evaluation: three representative models – DeepLog, DeepTraLog, and TraceVAE – are trained and tested on the same TrainTicket dataset comprising 106,312 traces from 35 microservices. Results that originate from our experiments are explicitly distinguished from those reported in prior work. This mixed-methods strategy addresses a methodological gap identified in the existing literature, where cross-study comparisons of anomaly detection models are complicated by differences in datasets, preprocessing, hardware, and evaluation protocols.

The remainder of this paper is organized as follows. Section II provides background on microservice architectures and distributed tracing. Section III examines the structural awareness capabilities of both approaches. Section IV presents detection accuracy benchmarks, including our unified experimental setup and results. Section V analyzes computational efficiency considerations. Section VI discusses limitations, implications, and future directions, and Section VII concludes the paper.

2. Background and Related work

A. Microservice Architecture and Observability

Microservice architecture represents a fundamental departure from traditional monolithic application design. In a monolithic system, all functional components share a single codebase and are deployed as a unified artifact. While this approach simplifies initial development and deployment, it creates significant challenges for scaling, maintenance, and technology evolution [11]. Microservice architecture addresses these limitations by decomposing applications into small, independently deployable services, each responsible for a specific business capability [12].

However, this decomposition comes at the cost of increased operational complexity. A single user request may traverse dozens or even hundreds of services before completion, with each service potentially introducing its own failure modes, latency characteristics, and resource constraints [13]. The challenge of understanding system behavior in this context-commonly referred to as "observability" – has become a critical concern for organizations operating at scale.

B. Distributed Tracing Fundamentals

Distributed tracing provides a mechanism for tracking request flow across service boundaries. The core abstraction is the "trace," which represents the complete journey of a request through the system. Each trace consists of multiple "spans," where each span represents a named, timed operation within a single service [7]. Spans are organized hierarchically, with parent-child relationships indicating invocation patterns.

Modern tracing systems propagate context information (typically including a trace ID and span ID) through request headers, enabling the reconstruction of complete request paths even in highly distributed environments. This contextual information, combined with timing data and service metadata, provides the foundation for various analytical approaches including anomaly detection, performance optimization, and root cause analysis [14], [15].

C. Traditional Anomaly Detection Approaches

Early approaches to anomaly detection in distributed systems relied primarily on statistical methods and rule-based systems. These approaches typically involved setting thresholds for metrics such as response time, error rates, or resource utilization. While effective for detecting gross violations, such approaches struggle with the dynamic and context-dependent nature of modern microservice systems. The advent of machine learning has enabled more sophisticated detection methods capable of learning complex patterns from historical data.

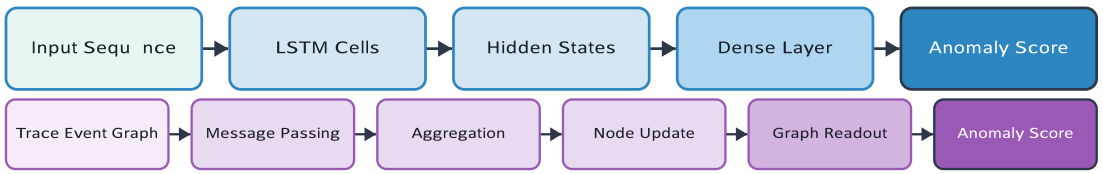
3. Structural awareness: Sequences vs. Hierarchies

The fundamental difference between sequence-based and graph-based models lies in their perception and representation of trace structure. This distinction has profound implications for anomaly detection capability, particularly in systems with complex invocation patterns.

A. Sequence-based Modeling with LSTM Networks

Traditional sequence models such as DeepLog and Multimodal LSTM treat traces as linear sequences of service invocations or log events [16], [17]. These models operate under the assumption of a sequential temporal flow where event A leads to event B in a strictly ordered manner. Long Short-Term Memory networks are particularly well-suited for this paradigm due to their ability to capture long-range dependencies through their gating mechanisms [18].

Figure 1
Architectural comparison: (a) Sequence-based LSTM pipeline, (b) Graph-based GNN pipeline



However, microservice traces are inherently tree-structured rather than linear [19]. This structural mismatch creates several fundamental limitations for sequence-based approaches:

- Invocation Hierarchy: A parent span may initiate multiple child spans simultaneously, representing parallel execution paths. Sequence models cannot naturally represent this branching structure [20].

- Asynchronous and Parallel Calls: Microservices frequently utilize parallel execution to reduce latency. In a sequence-based model, log events from parallel spans interleave in arbitrary orders, leading the model to falsely report unseen subsequences as anomalies [21].

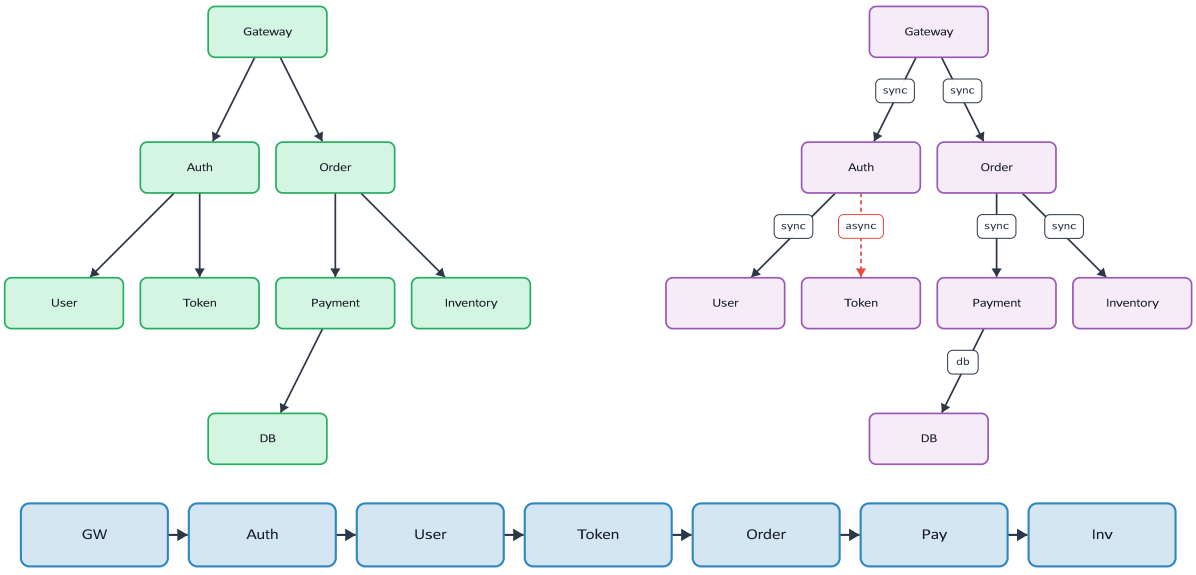
- Context Loss: When flattening a tree structure into a sequence, critical contextual information about the relationship between operations is lost, potentially masking important anomaly indicators.

B. Graph-based Modeling with Graph Neural Networks

Graph-based frameworks address these limitations by preserving the natural structure of distributed traces. Systems such as DeepTraLog utilize a Trace Event Graph (TEG) representation that unifies inter-service interactions (traces) and intra-service behaviors (logs) into a single cohesive structure [22]. In this paradigm, the trace is represented as a directed graph where nodes correspond to spans or log events and edges represent relationships between them.

Graph representations enable the explicit encoding of multiple relationship types, including: (1) Sequence relationships between consecutive events within a service; (2) Synchronous request relationships between caller and callee; (3) Synchronous response relationships for return paths; and (4) Asynchronous request relationships for fire-and-forget patterns [23]. This rich relational encoding provides models with the structural context necessary for accurate anomaly detection.

Figure 2
Trace representations: (a) Original tree structure (b) Graph with typed edges (structure preserved)., (c) Flattened sequence (information lost)



By modeling traces as graphs, GNNs-specifically Gated Graph Neural Networks (GGNNs)-can capture the "back-pressure" and cascading effects that propagate along service dependencies [24]. This capability is essential for detecting anomalies that manifest as structural deviations rather than simple metric threshold violations.

Advanced graph-based models like TraceVAE further extend this approach by utilizing a dualvariable architecture that encodes structural and temporal features separately [25]. This separation ensures that the model understands both the topological context of each service invocation and its temporal characteristics, enabling more nuanced anomaly detection.

4. Detection accuracy: Empirical evidence

Empirical evidence from multiple research studies indicates a significant performance gap between sequence-based and graph-based approaches for microservice anomaly detection. This section presents quantitative comparisons and discusses the underlying factors contributing to these differences.

A. Benchmark Methodology

The TrainTicket microservice system has emerged as a widely-used benchmark for evaluating anomaly detection approaches [26]. This system simulates a realistic ticket booking application comprising dozens of interconnected microservices. Researchers inject various fault types including service failures, latency anomalies, and resource exhaustion to evaluate detection capabilities across different anomaly categories.

B. Unified Experimental Setup

To address the methodological limitation of cross-study comparison, we conduct original experiments in which three representative models are evaluated under controlled conditions. We select DeepLog [16] as the sequence-based baseline, DeepTraLog [22] as the primary graph-based model, and TraceVAE [25] as an alternative graph-based architecture employing variational inference.

All models are trained and evaluated on traces extracted from the TrainTicket microservice benchmark [26], comprising 106,312 traces from 35 services (89,724 normal, 16,588 anomalous). The traces originate from 14 fault injection scenarios (F01–F14) covering service failures, latency anomalies, and resource exhaustion. We apply a consistent split: 62,806 training traces, 10,630 validation traces (8,972 normal + 1,658 anomalous), and 32,876 test traces (17,946 normal + 14,930 anomalous).

A unified data conversion pipeline transforms the raw traces into the input format required by each model. DeepLog and DeepTraLog experiments were conducted on an NVIDIA A100SXM4-40GB GPU; TraceVAE was evaluated on a Tesla T4 GPU due to dependency constraints of the DGL framework version required by its implementation. While this hardware difference limits direct comparison of computational efficiency between TraceVAE and the other models, detection accuracy metrics (Precision, Recall, F1) remain comparable as they are hardware-independent. We report all metrics using identical evaluation procedures across models.

C. Quantitative Performance Comparison

Table 1 presents comparative results from recent evaluation studies. As shown, graph-based models consistently outperform their sequence-based counterparts across precision, recall, and F1score metrics.

Table 1

Performance Comparison of Anomaly Detection Models. Results marked "This work" were obtained on 106,312 TrainTicket traces (62,806 train / 10,630 val / 32,876 test). DeepLog and Deep-TraLog used NVIDIA A100-SXM4-40GB; TraceVAE used Tesla T4 due to framework constraints.

Literature results are reported as published

Model	Type	Dataset	Hardware	P	R	F1	Source
TraceVAE	GNN+VAE	TrainTicket	Tesla T4	0.940	0.855	0.896	This work
DeepTraLog	GNN (GGNN)	TrainTicket	A100-40GB	0.937	0.807	0.868	This work
DeepLog	LSTM	TrainTicket	A100-40GB	0.688	0.844	0.758	This work
DeepTraLog	GNN (GGNN)	TrainTicket	2×V100	0.930	0.978	0.954	[22]
TraceVAE	GNN+VAE	TrainTicket	N/R	0.912	0.956	0.933	[25]

Continuation of the table

Model	Type	Dataset	Hardware	P	R	F1	Source
TCN-VAE	Hybrid	TrainTicket	N/R	0.938	0.927	0.933	[34]
Eadro	Multi-modal	TrainTicket	N/R	0.891	0.912	0.901	[34]
GRU-SVDD	Sequence	TrainTicket	N/R	0.834	0.803	0.818	[22]
Sleuth	Graph	N/R	N/R	0.856	0.789	0.821	[36]
DeepLog	Sequence	HDFS/BGL	N/R	0.762	0.721	0.741	[16]
LogAnomaly	Sequence	HDFS/BGL	N/R	0.724	0.698	0.711	[19]
MLP Baseline	Vector	TrainTicket	N/R	0.681	0.645	0.663	[25]
TraceAnomaly	Vector	TrainTicket	N/R	0.412	0.267	0.321	[9]

Figure 3
Performance comparison: (a) Detection accuracy metrics, (b) F1-score ranking by model type

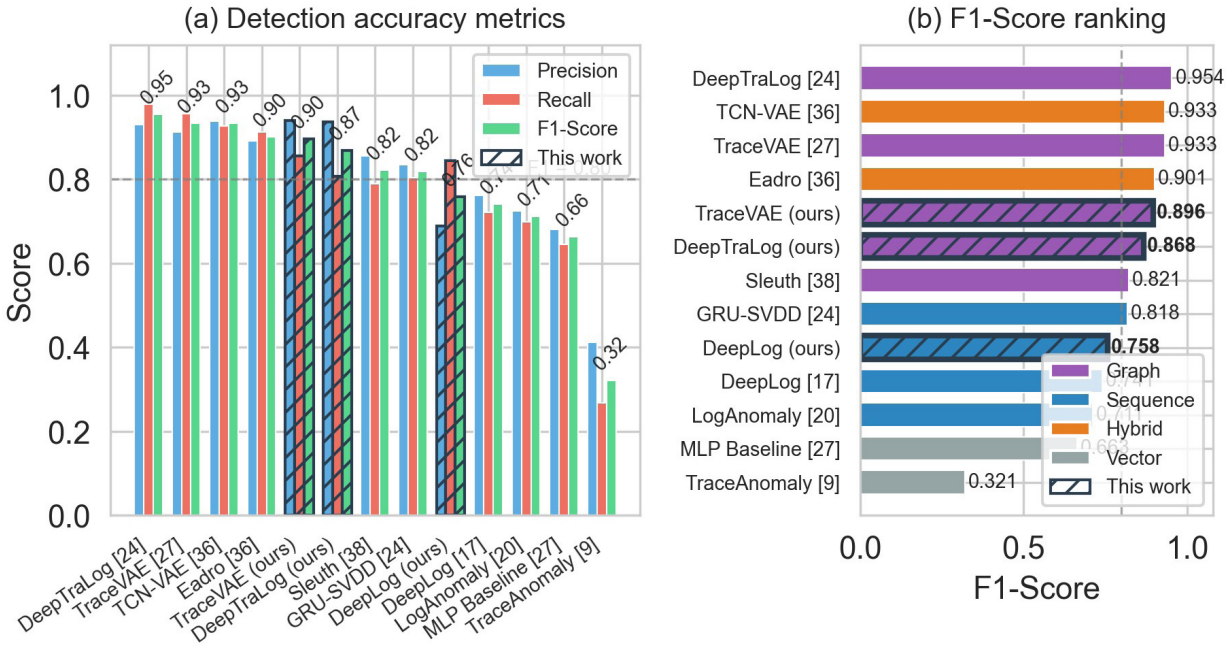
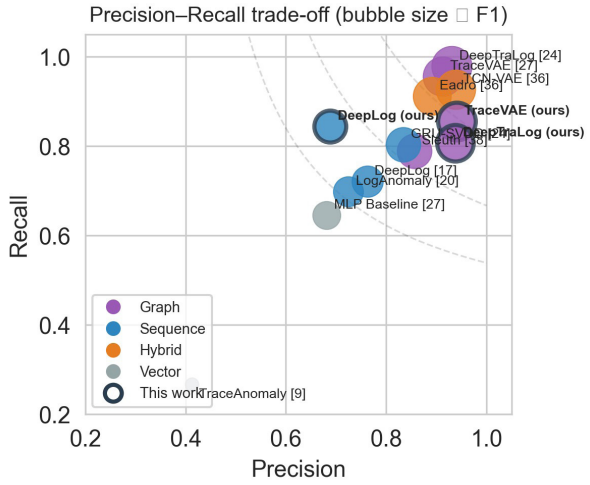


Table 1 presents both our experimental results and findings from the literature. In our unified evaluation, TraceVAE achieved the highest F1-score (0.896), followed by DeepTraLog (0.868), both substantially outperforming the LSTM-based DeepLog (0.758). Notably, our results differ from those reported in the original studies – for example, DeepTraLog achieves F1=0.954 in [22] versus 0.868 in our experiments – illustrating the sensitivity of these models to data splits, preprocessing pipelines, and hardware configurations. This discrepancy reinforces the need for unified benchmarking as presented in this work.

Graph-based models like TraceVAE address advanced probabilistic issues that particularly affect sequence-based approaches. One such issue is Negative Log-Likelihood (NLL) Inversion, where sequential models suffer from an "entropy gap" [25]. In this phenomenon, lower-dimensional anomalies-such as a missing node in a large trace-paradoxically result in lower anomaly scores than normal traces, leading to false negatives.

Figure 4
Precision–Recall trade-off across all evaluated models. Bubble size is proportional to F1score; bold edges indicate results from this work



D. Addressing Probabilistic Challenges

GNN-based approaches mitigate this issue through techniques such as Node Count Normalization and Gaussian Std-Limit within the graph framework [25], [27]. These methods effectively normalize for trace complexity, ensuring that anomalies are detected regardless of trace size. This capability is particularly important in production environments where trace complexity varies significantly across different request types.

5. Computational Efficiency Analysis

Microservice telemetry is increasingly high-dimensional, with individual traces potentially involving hundreds of service calls and thousands of log events [1]. Computational efficiency is therefore a critical consideration for production deployment of anomaly detection systems.

A. Training and Inference Latency

Empirical analysis reveals an interesting trade-off between training time and testing efficiency. In our unified experiments on the TrainTicket benchmark (NVIDIA A100 GPU), DeepTraLog processed the test set of 32,876 traces in 13.2 seconds compared to 3,115.1 seconds for DeepLog, yielding a 237× difference in inference time. This substantial speedup, obtained under controlled conditions with identical data and hardware, exceeds the approximately 40× improvement reported in prior cross-study comparisons [22] and suggests that the efficiency advantage of graph-based inference may be more pronounced than previously documented. GGNNs treat events as parallel network units amenable to efficient message-passing algorithms, whereas LSTMs must serially process every event in a sequence [28], [29].

Table 2
Training and Inference Time Comparison. Inference times are for the full test set (32,876 traces for "This work" rows). Times are not directly comparable across different hardware

Model	Type	Hardware	Training Time	Inference Time	Source
DeepTraLog	GNN (GGNN)	A100-40GB	1,618 s (~27 min)	13.2 s	This work
TraceVAE	GNN+VAE	Tesla T4	27,112 s (~7.5 h)	281.9 s	This work
DeepLog	LSTM	A100-40GB	7,584 s (~2.1 h)	3,115.1 s	This work
DeepTraLog	GNN (GGNN)	2×V100	~67 min	77 s	[22]
TraceVAE	GNN+VAE	N/R	~85 min	92 s	[25]
LogAnomaly	Sequence	N/R	~45 min	3,136 s	[19]
DeepLog	Sequence	N/R	~25 min	~800 s	[16]

Figure 5
Computational efficiency comparison: (a) Training time, (b) Inference time showing 237× speedup for DeepTraLog vs DeepLog

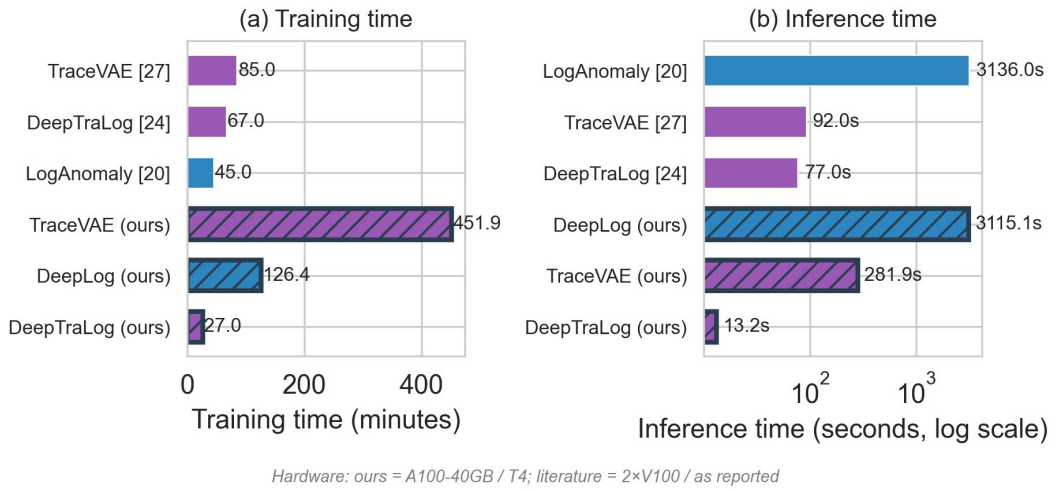


Figure 6
AUC-ROC comparison for models evaluated in this work

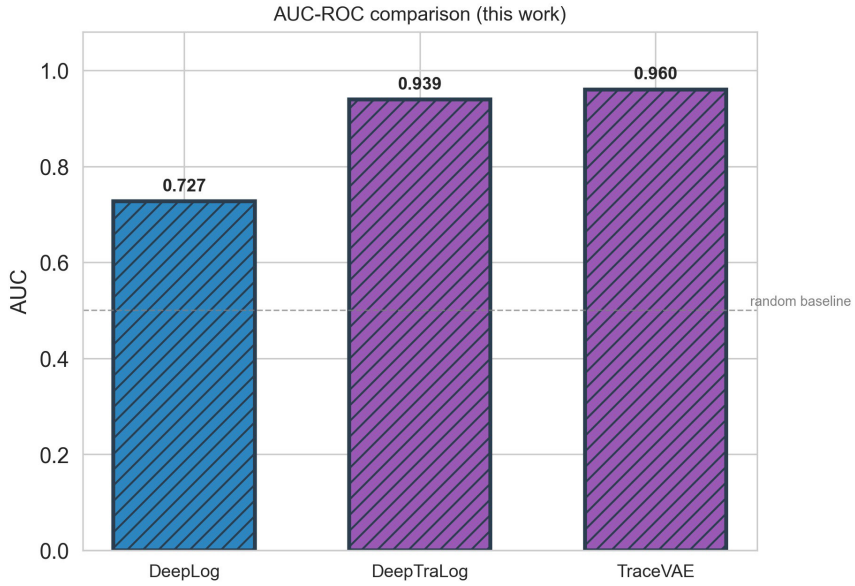


Table 3
Model Characteristics

Capability	DeepTraLog	TraceVAE	DeepLog	LogAnomaly
Structural Awareness	Full	Full	None	None
Parallel Call Handling	Yes	Yes	No	No
Log+Trace Integration	Combined	Trace only	Log only	Log only
Unsupervised	No	Yes	No	Yes
Interpretable	Partial	Limited	Partial	Limited

B. Scalability and Dimensionality Challenges

Sequence models that rely on sliding window approaches suffer from the "curse of dimensionality" when the number of distinct log or span event types exceeds typical limits [30]. As the event vocabulary grows, the model's ability to generalize degrades, and memory requirements increase polynomially.

Graph models mitigate this challenge through several mechanisms. First, soft-attention mechanisms allows the model to dynamically focus on nodes contributing most to an anomaly, effectively reducing the effective dimensionality of the problem [31]. Second, message-passing architecture of GNNs enables efficient information aggregation across graph neighborhoods without requiring explicit enumeration of all possible event sequences. These properties allow graph-based systems to scale effectively even as the number of events in a trace reaches 800–900 or more [25], [32].

2. Discussion

A. Practical Implications

The findings presented in this analysis have significant implications for practitioners designing observability systems for microservice architectures. The superior performance of graph-based approaches suggests that organizations should prioritize these methods for new deployments, particularly in systems with complex service dependencies and parallel execution patterns.

However, the choice between approaches should also consider organizational factors. Sequencebased models may remain appropriate for simpler architectures with predominantly linear call patterns, or in environments where training time constraints outweigh inference performance requirements. Additionally, the interpretability of sequence models may provide advantages in certain debugging and root cause analysis scenarios.

B. Limitations of Graph-Based Approaches

While graph-based models demonstrate clear advantages in structural awareness and detection accuracy, several practical limitations warrant consideration.

First, GNN-based methods are sensitive to incomplete or malformed trace data. In production environments, spans may be lost due to sampling policies, network partitions, or instrumentation failures, resulting in partial graphs. Unlike sequence models that degrade gracefully when elements are missing from a linear sequence, graph models may propagate errors through message-passing layers, amplifying the effect of missing nodes or edges on downstream representations [23].

Second, evolving microservice topologies introduce a cold-start challenge. When new services are deployed or existing call paths change, the graph structure diverges from patterns observed during training. Retraining or fine-tuning the model becomes necessary, and frequency of architectural changes in actively developed systems may make this impractical without automated retraining pipelines [1].

Third, training cost varies significantly across architectures. In our experiments, TraceVAE required 27,112 seconds of training time compared to 1,618 seconds for DeepTraLog and 7,584 seconds for DeepLog. The variational inference mechanism in TraceVAE, while yielding the highest detection accuracy, imposes a substantial computational overhead during training that must be weighed against the marginal F1 improvement over DeepTraLog (0.896 vs. 0.868).

Fourth, the explainability of GNN-based anomaly detection remains limited. While methods such as GNNExplainer [33] offer post-hoc interpretation of node-level predictions, generating humanreadable explanations for why a particular trace was flagged as anomalous is an open challenge. For operational teams, understanding the root cause behind an alert is as important as the detection itself [37].

Finally, deployment complexity is higher for graph-based systems. Constructing graph representations from raw traces at serving time requires additional preprocessing step that introduces latency and engineering effort compared to simpler tokenization required by sequence models.

C. Future Research Directions

Several promising research directions emerge from this analysis. First, hybrid architectures that combine the temporal modeling strengths of recurrent networks with the structural awareness of GNNs warrant further investigation. Second, the application of more advanced GNN variants, such as Graph Attention Networks (GATs) and Graph Transformers, to microservice anomaly detection remains underexplored. Third, the development of explainable anomaly detection methods that can identify not just that an anomaly occurred, but specifically which service interactions contributed to the anomaly, represents an important direction for operational utility [33]. Additionally, TraceGra [35] represents an existing mixed-methods strategy combining GNN and LSTM for simultaneous topology and temporal

pattern capture. Cross-system transfer learning and meta-learning approaches [42] offer a promising direction to reduce cold-start issues for new deployments. Recent comprehensive surveys [38], [39], [40] highlight the need for standardized benchmarks to enable fair cross-method comparison. Real-time distributed tracing anomaly detection in cloud environments [41] is emerging as an operational requirement. The precision-recall trade-off observed in our experiments – where GNNs favor precision while LSTMs favor recall – warrants further investigation into ensemble or cascading approaches.

7. Conclusion

This paper has compared sequence-based and graph-based deep learning approaches for anomaly detection in microservice traces through both a structured literature analysis and original experiments on the TrainTicket benchmark.

The experimental results confirm that graph-based models capture structural properties of distributed traces more effectively than sequential alternatives. In our unified evaluation, TraceVAE achieved the highest F1-score of 0.896, followed by DeepTraLog at 0.868, while the LSTM-based DeepLog reached 0.758. These findings are consistent in direction with results reported in prior independent studies, though absolute F1 values differ due to variations in data splits, preprocessing, and evaluation conditions – illustrating precisely why unified benchmarking is necessary.

A notable finding is the scale of inference efficiency gains. DeepTraLog processed the test set 237 times faster than DeepLog on identical hardware, substantially exceeding the approximately 40× speedup reported in cross-study comparisons. At the same time, our experiments reveal that graphbased models exhibit higher precision but lower recall than the LSTM baseline, suggesting that sequence models may be more sensitive to anomalies at the cost of more false positives.

These observations suggest that the choice between approaches depends on the operational context. For systems where precision is prioritized and structural fidelity matters – such as production environments with stable topologies and complete instrumentation – graph-based models offer measurable advantages. For rapidly evolving systems where high recall is critical and computational resources are constrained, sequence-based models may remain a pragmatic alternative.

Future work should investigate hybrid architectures that combine temporal and structural modeling, the application of Graph Transformers to trace analysis, and methods for generating interpretable explanations from GNN-based detectors.

Funding

The study was funded by the Science Committee of the Ministry of Science and Higher Education of the Republic of Kazakhstan, target grant # BR28713313.

Conflicts of Interest

The authors declare no conflict of interest.

Author Contributions

Conceptualization, Y.Z. and A.Z.; Methodology, Y.Z., A.Z. and S.M; Software, Y.Z.; Validation, Y.Z., A.Z., A.A. and S.M; Formal Analysis, Y.Z., A.Z., A.A.. and S.M; Investigation, Y.Z., A.Z. and A.A; Resources, Y.Z. and A.Z.; Data Curation, Y.Z., A.Z. and S.M; Writing – Original Draft Preparation, Y.Z. and A.Z.; Writing – Review & Editing, Y.Z. and A.Z.; Visualization, Y.Z. and A.Z.; Supervision, A.Z.; Project Administration, A.Z.; Funding Acquisition, A.Z.

References

1. X. Zhou, X. Peng, T. Xie, J. Sun, C. Ji, W. Li, and D. Ding, "Fault analysis and debugging of microservice systems: Industrial survey, benchmark system, and empirical study," *IEEE Trans. Softw. Eng.*, vol. 47, no. 2, pp. 243–260, 2021.

2. Y. Gan, Y. Zhang, D. Cheng, A. Shetty, P. Rathi, N. Katarki, A. Bruno, J. Hu, B. Ritchken, B. Jackson, K. Hu, M. Pancholi, Y. He, P. Clancy, C. Colen, F. Wen, C. Leung, S. Wang, L. Zaruvinsky, M. Espber, R. Lin, Z. Liu, J. Padilla, and C. Delimitrou, "Seer: Leveraging Big Data to Navigate the Complexity of Performance Debugging in Cloud Microservices," in *Proc. ASPLOS*, 2019, pp. 19–33.

3. L. Wu, J. Tordsson, E. Elmroth, and O. Kao, "MicroRCA: Root Cause Localization of Performance Issues in Microservices," in *Proc. IEEE/IFIP NOMS*, 2020, pp. 1–9.

4. S. Newman, *Building Microservices: Designing Fine-Grained Systems*. O'Reilly Media, 2015.

5. A. Brandón, M. Solé, A. Huélamo, D. Solans, M. S. Pérez, and V. Muntés-Mulero, "Graph-based root cause analysis for service-oriented and microservice architectures," *J. Syst. Softw.*, vol. 159, 2020, Art. no. 110432.

6. D. Liu, C. He, X. Peng, F. Lin, C. Zhang, S. Gong, Z. Li, J. Ou, and Z. Wu, "MicroHECL: HighEfficient Root Cause Localization in Large-Scale Microservice Systems," *arXiv:2103.01782*, 2021.

7. B. H. Sigelman, L. A. Barroso, M. Burrows, P. Stephenson, M. Plakal, D. Beaver, S. Jaspan, and C. Shanbhag, "Dapper, a Large-Scale Distributed Systems Tracing Infrastructure," Google, Tech. Rep., 2010.

8. P. Mace and R. Fonseca, "Universal Context Propagation for Distributed System Instrumentation," in *Proc. EuroSys*, 2018, pp. 8:1–8:18.

9. P. Liu, H. Xu, Q. Ouyang, R. Jiao, Z. Chen, S. Zhang, J. Yang, L. Mo, J. Zeng, W. Xue, and D. Pei, "Unsupervised Detection of Microservice Trace Anomalies through Service-Level Deep Bayesian Networks," in *Proc. ISSRE*, 2020, pp. 48–58.

10. S. He, J. Zhu, P. He, and M. R. Lyu, "Loghub: A Large Collection of System Log Datasets towards Automated Log Analytics," *arXiv:2008.06448*, 2020.

11. N. Dragoni, S. Giallorenzo, A. L. Lafuente, M. Mazzara, F. Montesi, R. Mustafin, and L. Safina, "Microservices: yesterday, today, and tomorrow," in *Present and Ulterior Software Engineering*, 2017, pp. 195–216.

12. J. Thönes, "Microservices," *IEEE Softw.*, vol. 32, no. 1, pp. 116–116, 2015.

13. M. Fowler and J. Lewis, "Microservices: A definition of this new architectural term," *martinfowler.com*, 2014. [Online]. Available: https://martinfowler.com/articles/microservices. html

14. R. Sambasivan, R. Fonseca, I. Shafer, and G. R. Ganger, "So, you want to trace your distributed system? Key design insights from years of practical experience," Carnegie Mellon University, Tech. Rep., 2014.

15. J. Kaldor, J. Mace, M. Bejda, E. Gao, W. Kuropatwa, J. O'Neill, K. W. Ong, B. Schaller, P. Shan, B. Viscomi, V. Venkataraman, K. Veeraraghavan, and Y. J. Song, "Canopy: An End-to-End Performance Tracing And Analysis System," in *Proc. SOSP*, 2017, pp. 34–50.

16. M. Du, F. Li, G. Zheng, and V. Srikumar, "DeepLog: Anomaly Detection and Diagnosis from System Logs through Deep Learning," in *Proc. ACM CCS*, 2017, pp. 1285–1298.

17. S. Nedelkoski, J. Cardoso, and O. Kao, "Anomaly Detection from System Tracing Data Using Multimodal Deep Learning," in *Proc. IEEE CLOUD*, 2019, pp. 179–186.

18. S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

19. W. Meng, Y. Liu, Y. Zhu, S. Zhang, D. Pei, Y. Liu, Y. Chen, R. Zhang, S. Tao, P. Sun, and R. Zhou, "LogAnomaly: Unsupervised Detection of Sequential and Quantitative Anomalies in Unstructured Logs," in *Proc. IJCAI*, 2019, pp. 4739–4745.

20. A. Brown, A. Tuor, B. Hutchinson, and N. Nichols, "Recurrent Neural Network Attention Mechanisms for Interpretable System Log Anomaly Detection," in *Proc. AMLC*, 2018, pp. 1–8.

21. Z. Chen, J. Liu, W. Gu, Y. Su, and M. R. Lyu, "Experience Report: Deep Learning-based System Log Analysis for Anomaly Detection," *arXiv:2107.05908*, 2021.

22. C. Zhang, X. Peng, C. Sha, K. Zhang, Z. Fu, X. Wu, Q. Lin, and D. Zhang, "DeepTraLog: Tracelog combined microservice anomaly detection through graph-based deep learning," in *Proc. ICSE*, 2022, pp. 623–634.

23. H. X. Nguyen, S. Zhu, and M. Liu, "A Survey on Graph Neural Networks for MicroserviceBased Cloud Applications," *Sensors*, vol. 22, no. 23, 2022, Art. no. 9492.

24. Y. Li, D. Tarlow, M. Brockschmidt, and R. Zemel, "Gated Graph Sequence Neural Networks," *arXiv:1511.05493*, 2015.

25. Z. Xie, H. Chen, R. Cheng, Q. Lu, Y. Wang, L. Wang, H. Chen, and M. Yang, "Unsupervised Anomaly Detection on Microservice Traces through Graph VAE," in *Proc. WWW*, 2023, pp. 2874–2884.

26. X. Zhou, X. Peng, T. Xie, J. Sun, C. Xu, C. Ji, and W. Zhao, "Poster: Benchmarking microservice systems for software engineering research," in *Proc. ICSE-C*, 2018, pp. 323–324.

27. D. P. Kingma and M. Welling, "Auto-Encoding Variational Bayes," *arXiv:1312.6114*, 2013.

28. T. N. Kipf and M. Welling, "Semi-Supervised Classification with Graph Convolutional Networks," *arXiv:1609.02907*, 2016.

29. J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl, "Neural Message Passing for Quantum Chemistry," in *Proc. ICML*, 2017, pp. 1263–1272.

30. R. Bellman, "Dynamic Programming," *Science*, vol. 153, no. 3731, pp. 34–37, 1966.

31. P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph Attention Networks," *arXiv:1710.10903*, 2017.

32. W. L. Hamilton, R. Ying, and J. Leskovec, "Inductive Representation Learning on Large Graphs," in *Proc. NeurIPS*, 2017, pp. 1024–1034.

33. Z. Ying, D. Bourgeois, J. You, M. Zitnik, and J. Leskovec, "GNNExplainer: Generating Explanations for Graph Neural Networks," in *Proc. NeurIPS*, 2019, pp. 9244–9255.

34. C. Lee, T. Yang, Z. Chen, Y. Su, and M. R. Lyu, "Eadro: An End-to-End Troubleshooting Framework for Microservices on Multi-source Data," in *Proc. ICSE*, 2023, pp. 1750–1762.

35. J. Chen, F. Liu, G. Zhong, J. Jiang, D. Xu, Z. Tan, and S. Shi, "TraceGra: A trace-based anomaly detection for microservice using graph deep learning," *Computer Communications*, vol. 204, pp. 109–117, 2023.

36. S. Zhang, P. Jin, Z. Lin, Y. Sun, B. Zhang, S. Xia, Z. Li, Z. Zhong, M. Ma, W. Jin, Z. Zhang, D. Zhu, and D. Pei, "Robust Failure Diagnosis of Microservice System Through Multimodal Data," *IEEE Trans. Serv. Comput.*, vol. 16, no. 5, pp. 3537–3550, 2023.

37. G. Yu, P. Chen, Y. Li, H. Chen, X. Li, and Z. Zheng, "Nezha: Interpretable Fine-Grained Root Causes Analysis for Microservices on Multi-modal Observability Data," in *Proc. ESEC/FSE*, 2023, pp. 553–565.

38. Z. Z. Darban, G. I. Webb, S. Pan, C. C. Aggarwal, and M. Salehi, “Deep Learning for Time Series Anomaly Detection: A Survey,” *ACM Comput. Surv.*, vol. 57, no. 1, pp. 1–42, 2024.

39. T. Wang and G. Qi, “A Comprehensive Survey on Root Cause Analysis in (Micro) Services: Methodologies, Challenges, and Trends,” *arXiv:2408.00803*, 2024.

40. S. Zhang, S. Xia, W. Fan, B. Shi, X. Xiong, Z. Zhong, M. Ma, Y. Sun, and D. Pei, “Failure Diagnosis in Microservice Systems: A Comprehensive Survey and Analysis,” *arXiv:2407.01710*, 2024.

41. M. Raeiszadeh, A. Ebrahimzadeh, A. Saleem, R. Glitho, J. Eker, and R. A. F. Mini, “Real-Time Anomaly Detection Using Distributed Tracing in Microservice Cloud Applications,” in *Proc. IEEE CloudNet*, 2023, pp. 1–7.

42. Y. Wang, M. Mäntylä, J. Nyyssölä, K. Ping, and L. Wang, “Cross-System Software Log-based Anomaly Detection Using Meta-Learning,” in *Proc. SANER*, 2025, pp. 1–12.

Information about the authors:

Yesset Zhussupov is a PhD student in Software Engineering at L.N. Gumilyov Eurasian National University. He received his MSc in Software Engineering from the same university in 2022. Mr. Zhussupov serves as Chief Technology Officer at an AFSA-licensed brokerage organization, where he architects systems integrating big data and machine learning technologies. His research focuses on neural network applications in finance and economics, with particular emphasis on market forecasting and predictive modeling. His recent work includes the paper “Study of the Possibilities of Using Deep Artificial Intelligence in Forecasting the Green Paper Market in Kazakhstan.” In 2015, he received the Intel Excellence Award for his work applying neural networks to computer vision–based emotion recognition (Almaty, Kazakhstan, e-mail: yesset.zhussupov@gmail.com. ORCID iD: 0009-0001-0946-8094).

*PhD Dr. Ainur Zhumadillayeva is an Associate Professor of Associate Professor of Computer and Software Engineering at L.N. Gumilyov Eurasian National University. She received her PhD degree in the specialty *05.13.18 – Mathematical Modeling, Numerical Methods, and Software Packages* from the Institute of Mathematics of the Ministry of Education and Science of the Republic of Kazakhstan (Almaty) in 2010. Ainur Zhumadillayeva is the author and coauthor of educational publications and scientific monographs and has published more than 60 scientific and educational-methodical works in Kazakhstan and abroad, including publications indexed in international databases such as Scopus and Clarivate Analytics. Her research interests include mathematical modeling, numerical methods, machine learning, and artificial neural networks. She is a Member of the Institute of Electrical and Electronics Engineers (IEEE) (Astana, Kazakhstan. ORCID iD: 0000-0003-1042-0415).*

Shona Shinassylov is a Researcher and Lecturer at the Faculty of Information Technology and Artificial Intelligence, Al-Farabi Kazakh National University. He received his Master’s degree in Information Technology from Al-Farabi Kazakh National University and completed his doctoral studies in 2025. His research interests include digital twins, artificial intelligence, machine learning, deep reinforcement learning, smart buildings, the Internet of Things (IoT), energy management systems, and intelligent decision-support systems. His current research focuses on the design and development of virtual prototypes of smart building energy management systems based on digital twin technologies, machine learning methods, and multi-objective reinforcement learning (Almaty, Kazakhstan ORCID iD: 0009-0002-5491-7255).

Ainur Amirtayeva is a Master’s student in the dual-degree program between Al-Farabi Kazakh National University (KazNU) and Northwestern Polytechnical University (NWPU). She received her Bachelor’s degree in Information Security Systems from Al-Farabi Kazakh National University. Her academic and professional interests include cybersecurity, artificial intelligence, machine learning, deep learning, neural networks, parallel and distributed programming, phishing detection, computer vision, network traffic analysis, intrusion detection systems, and the application of AI in healthcare. Her research focuses on the development of intelligent information security systems, the application of neural networks, and the use of parallel computing techniques to improve machine learning algorithms (Almaty, Kazakhstan, e-mail: amirtayevaainur@gmail.com. ORCID iD: 0009-0008-4579-287X).

Submission received: 17 January, 2026
Revised: 28 April, 2026
Accepted: 4 May, 2026

IRSTI 28.23.37 https://doi.org/10.26577/jpcsit4220266

Tin Trung Chau , Miras Shaltayev , Kulash Talapiden ,
Muhammad Auwal Shehu , Ahmad Bala Alhassan*
Nazarbayev University, Astana, Kazakhstan
*e-mail: ahmad.alhassan@nu.edu.kz

ENHANCING WIND SPEED FORECASTING WITH LARGE LANGUAGE MODELS: A CASE STUDY OF KAZAKHSTAN

Abstract. Accurate wind speed forecasting plays an important role in the planning, operation, and control optimization of modern wind energy systems. Kazakhstan, the country with the largest wind energy potential in Central Asia, is facing significant challenges due to the highly variable, nonlinear, and non-stationary nature of wind. This study proposes a wind-speed forecasting framework based on frozen pre-trained Transformer backbones (BERT/BART) with lightweight projection-based adaptation for time-series inputs. The study employs frozen pre-trained Transformer backbones with self-attention and lightweight projection-based adaptation of Bidirectional Encoder Representations from Transformers (BERT) and Bidirectional and Auto-Regressive Transformers (BART) as the core of the model. In this retrospective analysis, hourly wind speed data from five representative cities across Kazakhstan are used to evaluate simulated operational forecast performance at seven time steps: 1, 3, 6, 36, 72, 144, and 432 hours. The LLMs were compared with AutoRegressive Integrated Moving Average (ARIMA), Segmented Recurrent Neural Network (SegRNN), and Patch Time Series Transformer (PatchTST) models, and the results showed superior accuracy and higher stability in long-term forecasting. These results support the potential of pre-trained Transformer backbones as effective sequence models for wind-speed forecasting under diverse climatic conditions.

Keywords: Wind energy forecasting, Large language models, Bidirectional Encoder Representations from Transformers, Bidirectional and Auto-Regressive Transformers, Time series forecasting.

1. Introduction

Kazakhstan is one of the countries with the largest wind energy potential in Central Asia, with a vast territory characterized by steppe and desert terrain. Approximately 50% of Kazakhstan's territory is suitable for wind energy development, including Aktobe, Aktau, Mangystau, and Astana, with average wind speeds of 7.5 m/s at altitudes of 50–100 meters [1]. According to international reports, Kazakhstan's total technical wind energy potential is estimated to reach 920 terawatt-hours (TWh) per year, significantly exceeding the country's current electricity demand. This indicates that wind power can be a key pillar of the energy transition, enabling carbon neutrality by 2060 [2], [3].

However, the exploitation of wind energy depends primarily on wind speed. In theory, wind power scales with the cube of wind speed, which means that even a small change in wind speed can have a significant impact on energy production [4], [5]. Meanwhile, wind speed in Kazakhstan is nonlinear, strongly seasonally fluctuates, and is influenced by topography, particularly in the coastal areas of the Caspian Sea and the eastern mountainous region [6]. This variability makes it challenging to plan, operate, and dispatch wind power systems, reducing the reliability of power generation and the efficiency of integration into the national grid.

In this context, the development of advanced wind speed forecasting models with high accuracy is a necessary step to improve forecasting accuracy and reliability. These forecasting models play a crucial role in the entire value chain of wind power systems, from planning and operation to control optimization [7], [8], [9]. Furthermore, accurate wind speed forecasting also helps to improve the stability and reliability of the power system, especially as the share of renewable energy in the grid increases. Modern forecasting models enable the optimization of turbine operations, reduce dispatch costs, and facilitate the integration of wind

power with other energy sources, such as solar power or storage systems [10]. As a result, Kazakhstan can enhance the efficiency of wind resource exploitation, ensure national energy security, and promote a sustainable transition to a low-carbon economy.

Given the importance of forecasting, this paper proposes an advanced wind speed forecasting framework that uses frozen pre-trained Transformer backbones to model temporal dependencies in wind-speed sequences. Unlike traditional statistical methods or models, pre-trained Transformer architectures provide strong sequence modeling capacity to model complex, nonlinear relationships and long-term temporal dependencies, which are crucial for capturing the stochastic, highly variable nature of wind speed [11].

The model is applied directly to wind speed data from several cities in Kazakhstan to evaluate its performance under different climatic and topographic conditions. The results are expected to assess whether frozen pre-trained Transformer backbones can effectively capture temporal variation patterns and improve both short- and long-term wind-speed forecasting accuracy. Furthermore, this approach contributes to improving the quality of wind energy forecasting in Kazakhstan. It highlights the potential application of pre-trained Transformer backbones in renewable energy forecasting and management.

Given the importance of accurate wind speed forecasting, numerous studies have been conducted to enhance the reliability and quality of forecasts. Traditional forecasting methods, such as ARIMA [12], Seasonal-ARIMA [13], and ARIMA-ANN-Kalman [14], are relatively easy to implement and straightforward models. These models are primarily based on strict linear assumptions, and the data must be stable and consistent [15]. Therefore, these models often struggle to model the strong nonlinear and random features inherent in wind data, especially in areas with complex topography. Additionally, traditional deep learning models, such as CNNs [16], RNNs [17], LSTMs [18], and GRUs [19], have demonstrated promising performance in time series forecasting. However, these deep learning models still have limitations in maintaining long-term dependencies and are also susceptible to vanishing gradients when processing long data series.

The advent of the Transformer architecture marked a significant step forward, thanks to its self-attention mechanism, which enables the modeling of global relationships in time series. Typical variants such as Informer [20], Autoformer [21], and FEDformer [22] have demonstrated superior performance in handling non-stationarity and long-term dependencies in wind data.

Based on this foundation, pre-trained Transformer backbones such as BERT [23], BART [24], GPT [25], and LLaMA [26] have been extended from natural language processing to time series forecasting, opening up new approaches for reusing pre-trained sequence models in non-text domains. These models are capable of capturing long-term dependencies and modeling complex nonlinear relationships in sequential data. Two main approaches explored in recent research include (1) Intramodal Transfer Learning (ITL), which is capable of directly fine-tuning pre-trained Transformer models on sequence data to learn deep temporal features, and (2) Cross-modal Knowledge Transfer (CKT), which allows converting sequence data into text format to leverage pre-trained model priors [27].

These qualities of LLMs are beneficial for wind speed forecasting. Wind data are non-stationary, subject to seasonal cycles, varied terrain, and random noise, making modeling complex. Traditional statistical and deep learning models often struggle to handle nonlinear variations and long-term dependencies in time series data.

In contrast, pre-trained Transformer backbones with global self-attention and contextual sequence representations can flexibly adapt to data fluctuations, learning both short-term and long-term trends in wind behavior. Thanks to parameter reuse and lightweight adaptation, these models can generalize across multiple regions or turbines, reducing training costs and increasing adaptability to local conditions.

Therefore, integrating frozen pre-trained Transformer backbones into wind speed forecasting represents a significant step forward, moving from purely data-driven regression models to pre-trained backbone-based sequence forecasting systems, thereby enhancing accuracy, reliability, and support for intelligent energy management in modern wind power systems.

Objectives and contributions

Based on the identified research gaps and challenges, this study proposes developing an pre-trained Transformer backbone-based wind-speed forecasting framework for Kazakhstan

meteorological data. The main contributions of the paper can be summarized as follows:

- Proposing a wind speed forecasting framework based on frozen pre-trained Transformer backbones with a self-attention mechanism and fixed pre-trained sequence modeling capability to model nonlinearity, non-stationarity, and long-term dependence in wind data.
- Applying and validating the model on real-world wind datasets in many cities of Kazakhstan to reflect diverse climatic and topographic conditions, thereby evaluating the model's generalization ability and stability.
- Comparing the performance of the proposed frozen pre-trained Transformer backbones with current advanced models to evaluate the empirical accuracy and reliability of the proposed method.
- Providing a reference framework for time series and renewable energy forecasting applications, contributing to grid operation optimization, and with potential relevance to Kazakhstan's carbon neutrality strategy.

2. Materials and methods

2.1 Dataset and preprocessing

The study uses hourly wind speed data collected from NASA's Prediction of Worldwide Energy Resources (POWER) project. Wind speed data were collected from January 2010 to July 2025 at hourly intervals over cities including Aktobe, Aktau, Kokshetau, Taldykorgan, and Uralsk. These regions were selected to reflect the diversity of climate and topography, from coastal to inland and mountainous areas. Each dataset consists of time-labeled wind speed time series. Data were collected at a height of 80 m, corresponding to the typical hub height of wind turbines. Figure 1 presents the location of five cities.

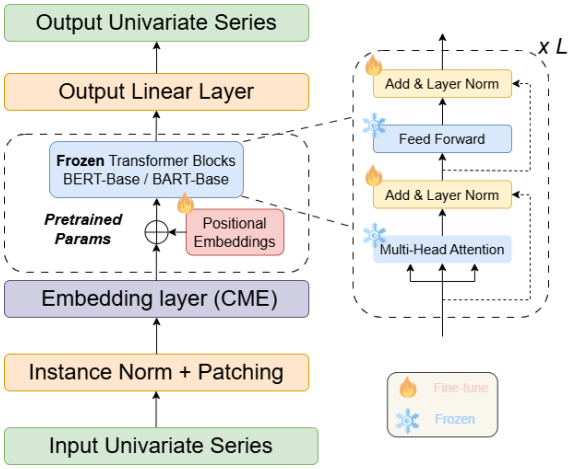
The data were split into training, test, and validation sets with proportions of 70%, 20%, and 10%, respectively. Subsequently, the data were cleaned and interpolated to handle missing or outlier values using cubic spline interpolation, and short-term noise was mitigated using Savitzky–Golay filtering. Crucially, the parameters for these filtering operations were fitted exclusively on the training set and then applied to the validation and test sets. Next, the data were standardized using Z-score normalization, with the mean and standard deviation computed from the training data only. Finally, the normalized time series was divided into sliding windows. This chronological processing converts NASA POWER data from raw meteorological data into a structured time series suitable for Transformer and LLM models, ensuring consistency, stability, and a rigorous out-of-sample evaluation.

Figure 1
Geographical locations of the five study sites in Kazakhstan

2.2 Model structure

The diagram in Figure 2 illustrates the model structure, including global Z-score standardization of the input sequence, patching, linear projection, and positional encoding. To improve training stability and ensure consistent scaling across datasets, the input sequences are first standardized using the global Z-score normalization described in Section 2.1, where the mean and standard deviation are estimated from the training set only. Then, it is passed through the pre-trained Transformer blocks (frozen) and combined with fine-tuning layers (Add & Layer Norm). The output is passed through a linear layer to create the predicted sequence. This structure leverages the long-range sequence modeling capacity of the frozen pre-trained Transformer blocks and reduces training costs by fixing most of the original model's parameters.

Figure 2
Architecture of LLM Reprogramming for Univariate Time Series Forecasting



Problem Definition. The problem of forecasting wind speed time series is formulated as follows:

Given a historical observation data series of length L, denoted by

$$\mathbf{V} = (v_1, v_2, \dots, v_L), \quad (1)$$

Where $v_t \in \mathbb{R}^M$ – a multivariate vector consisting of M features recorded at time t. In this study, only univariate wind speed is forecasted, so $M = 1$.

The goal of the model is to learn the time-dependent relationship in the input data series $\mathbf{V}$ to forecast a future value series of length T, denoted by

$$\hat{\mathbf{Y}} = (v_{L+1}, v_{L+2}, \dots, v_{L+T}). \quad (2)$$

In other words, the problem is to construct a prediction function $f(\cdot)$ such that

$$\hat{\mathbf{Y}} = f(\mathbf{V}),$$

to minimize the error between the forecast value $\hat{\mathbf{Y}}$ and the actual value $\mathbf{Y}$. In the context of this study, v_t represents the wind speed at time t, and the model must accurately predict the wind speed in the following T hours based on the observation data from the previous L hours.

Data Normalization. The observed wind speed series $\mathbf{V} = (v_1, v_2, \dots, v_L)$ is normalized to reduce the influence of amplitude fluctuations and stabilize the training process. The mean and standard deviation are calculated as follows:

$$\mu = \frac{1}{L}\sum_{t=1}^{L} v_t \,, \sigma = \sqrt{\frac{1}{L}\sum_{t=1}^{L}(v_t - \mu)^2 + \varepsilon}, \quad (3)$$

where

(ε) – a very small constant to avoid division by 0.

The series is normalized according to the formula:

$$\tilde{v}_t = \frac{v_t - \mu}{\sigma}. \quad (4)$$

The resulting series $\widetilde{\mathbf{V}} = (\tilde{v}_1, \tilde{v}_2, \dots, \tilde{v}_L)$has a mean of 0 and a variance of 1.

It should be noted that this study employs dataset-level standardization fitted on the training set and then applied consistently to the validation and test sets, rather than instance-wise normalization performed separately for each input window.

Patching. The normalized sequence$\widetilde{\mathbf{V}}$ is divided into patches to reduce the input length and increase the ability to capture the local structure in the time domain. With each patch length $\boldsymbol{P}$ and stride $\boldsymbol{S}$, the number of patches obtained is calculated as:

Each patch is denoted as:

$$N = \left\lfloor \frac{L-P}{S} \right\rfloor + 2, n = 1,2, \dots, N \quad (5)$$

The set of all patches forms a matrix$\mathbf{P} = [p_1, p_2, \dots, p_N]^\top \in \mathbb{R}^{N \times P}$. Patching helps the model to process in parallel and learn more effectively the local features of the sequence.

Embedding. Each patch$p_n$is projected into the hidden space of dimension d_{model} via a linear embedding layer:

$$\mathbf{E}_n = \mathbf{W}_e \mathbf{p}_n + \mathbf{b}_e,$$

where

($\mathbf{W}_e \in \mathbb{R}^{d_{\text{model}} \times P}$) – the learnable weight matrix

($\mathbf{b}_e \in \mathbb{R}^{d_{\text{model}}}$)- translation vector.

To preserve the temporal order information between patches, a positional encoding component is added to each embedding vector:

$$\mathbf{Z}_n = \mathbf{E}_n + PE_n,$$

where PE_n is defined as:

$$PE_{(n,2i)} = \sin\left(\frac{n}{10000^{\frac{2i}{d_{\text{model}}}}}\right), \quad (6)$$

$$PE_{(n,2i+1)} = \cos\left(\frac{n}{10000^{2i/d_{\text{model}}}}\right).$$

The result is the embedded signature string:

$$\mathbf{Z} = [\mathbf{Z}_1, \mathbf{Z}_2, \dots, \mathbf{Z}_N] \in \mathbb{R}^{N \times d_{\text{model}}}. \quad (7)$$

LLM Backbone. The embedding sequence $\mathbf{Z}$ is fed into a Transformer encoder consisting of L_t layers, each layer including a multi-head self-attention mechanism and a feed-forward network (FFN). The calculation formula for an attention head is defined as follows:

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{Softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d_k}}\right)\mathbf{V}, \quad (8)$$

with $\mathbf{Q} = \mathbf{Z}\mathbf{W}_Q, \mathbf{K} = \mathbf{Z}\mathbf{W}_K, \quad \mathbf{V} = \mathbf{Z}\mathbf{W}_V,$ and $\mathbf{W}_Q, \mathbf{W}_K, \mathbf{W}_V \in \mathbb{R}^{d_{\text{model}} \times d_k}$ are the learned weight matrices.

The results from all the heads are concatenated and projected linearly:

$$\mathbf{O} = \text{Concat}(\text{Att}_1, \dots, \text{Att}_H)\mathbf{W}_O.$$

Then, the residual connection block and FFN network are applied:

$$\mathbf{H}^{(l)} = \text{LayerNorm}\left(\mathbf{Z}^{(l-1)} + \text{MHA}\left(\mathbf{Z}^{(l-1)}\right)\right),$$
$$\mathbf{Z}^{(l)} = \text{LayerNorm}\left(\mathbf{H}^{(l)} + \text{FFN}(\mathbf{H}^{(l)})\right).$$

After the L_t layers, the final output of the encoder is:

$$\mathbf{H} = [\mathbf{H}_1, \mathbf{H}_2, \dots, \mathbf{H}_N] \in \mathbb{R}^{N \times d_{\text{model}}}.$$

This $\mathbf{H}$ representation contains contextual features and long-term dependencies in the wind speed series.

Forecasting Head. The output of the encoder $\mathbf{H}$ is passed through a linear regression layer to predict the future series. The projection is performed as follows:

$$\widehat{\mathbf{Y}} = \mathbf{H}\mathbf{W}_o + \mathbf{b}_o,$$

where $\mathbf{W}_o \in \mathbb{R}^{d_{\text{model}} \times T}$and $\mathbf{b}_o \in \mathbb{R}^T$.

The result is the forecast wind speed series in the following T time steps:

$$\widehat{\mathbf{Y}} = (\hat{v}_{L+1}, \hat{v}_{L+2}, \dots, \hat{v}_{L+T}).$$

In general, the entire forward propagation process is compactly represented as:

$$\hat{\mathbf{Y}} = f_{\text{Transformer}}(\text{Embed}(\tilde{\mathbf{V}})),$$

where

($f_{\text{Transformer}}(\cdot)$) – the nonlinear mapping of the large language model (LLM), including the embedding, self-attention, and linear regression layers.

Loss Function. The model is trained by minimizing the error between the predicted and actual wind speeds. The loss function used is **MSE**, which is defined as follows:

$$MSE = \frac{1}{NT}\sum_{i=1}^{N}\sum_{\tau=1}^{T}(v_{i,\tau} - \hat{v}_{i,\tau})^2, \quad (9)$$

where i indexes the training sample (or sliding window), and τ denotes the forecast step within the prediction horizon.

The optimization objective of the model is expressed as:

$$\theta^* = \text{argmin}_\theta MSE(\theta) \quad (10)$$

where

(θ) – the set of parameters of the model, including the weights of the embedding, attention, and regression layers.

The Adam algorithm updates weights with an adaptive learning rate, helping the training process converge quickly and stably. To avoid overfitting, dropout and early stopping techniques are used during training.

The linear embedding layer projects continuous time-series patches into the input space of the frozen Transformer backbone, enabling the reuse of pre-trained attention mechanisms for temporal dependency modeling. This allows the frozen Transformer blocks to serve as generalized pattern recognizers, leveraging their robust pre-trained attention mechanisms to capture complex temporal dependencies. For the BART model, specifically, only its encoder module was retained to match the unified encoder-only mathematical formulation described above, while its decoder was discarded.

3. Experiments

3.1 Datasets

In this study, five wind speed datasets from cities representing different regions were used to evaluate the performance of the LLMs. The experiments were designed to verify the model's forecast accuracy, its generalization across different climate domains, and its stability over the forecast horizon. The datasets covered both coastal and inland areas, reflecting the diversity of wind characteristics in space and time. The main information of the data is presented in Table 1.

Table 1
Datasets properties

Datasets	Location	Acquisition time	Sample rate	Samples
Aktobe	50.28 N, 57.17 E	Jan. 2010 – Jul. 2025	1 hour	136584
Aktau	47.12 N, 51.88 E	Jan. 2010 – Jul. 2025	1 hour	136584
Kokshetau	53.28 N, 69.38 E	Jan. 2010 – Jul. 2025	1 hour	136584
Taldykorgan	45.02 N, 78.38 E	Jan. 2010 – Jul. 2025	1 hour	136584
Uralsk	51.23 N, 51.34 E	Jan. 2010 – Jul. 2025	1 hour	136584

3.2. Evaluation and Benchmarking Setup

To evaluate the performance and generalization of the proposed wind speed forecasting model, the dataset is split into three non-overlapping parts: 70% for training, 20% for testing, and 10% for validation. All data are normalized using the mean and standard deviation calculated from the training set to ensure consistent scaling across experiments. The evaluation will focus on two main criteria: model accuracy and stability. Accuracy measures the deviation between the forecasted and actual values, while stability reflects the model's ability to adapt across different climate zones and forecast horizons. Standard evaluation metrics used include Mean Absolute Error (MAE) and Root Mean Square Error (RMSE), which are calculated by the formula:

$$\text{MAE} = \frac{1}{T}\sum_{t=1}^{T} | v_t - \hat{v}_t |, \text{RMSE} = \sqrt{\frac{1}{T}\sum_{t=1}^{T}(v_t - \hat{v}_t)^2}$$

To validate the effectiveness of the proposed LLM-based forecasting framework, the model performance is compared with a group of representtative baseline models. The baseline models include:

(1) ARIMA: a traditional autoregressive statistical model;

(2) SegRNN: a segmented recurrent network model for time series data;

(3) PatchTST: a Transformer model with a patch division mechanism specifically for time series forecasting.

In addition, two large language models (LLMs), including (4) BART and (5) BERT, are put into the experiment to evaluate the adaptability and feasibility of LLMs in wind speed forecasting.

For reproducibility and fair comparison, the main hyperparameter settings of the proposed models and all baseline methods are summarized in Table 2.

All models are trained and evaluated under the same experimental conditions to ensure fairness in comparison. Each experiment was repeated three times with different random seeds, and the mean and standard deviation of the evaluation indexes were reported to minimize the influence of random factors and ensure reproducibility.

Table 2
Main hyperparameter settings of the proposed models and baseline methods

Model	Input length	Patch / Segment length	Stride / No. segments	Hidden size	Main setting	Training
ARIMA	336	—	—	—	Dataset-specific *(p,d,q)* via minimum AIC on training data	—
SegRNN	336	48	7 segments	512	1-layer GRU	30 epochs
PatchTST	336	16	stride = 8	128	3 encoder layers, 4 heads	100 epochs
BERT/BART (proposed)	336	16	stride = 8	768	3 frozen layers, 12 heads	Adam, LR (=10^{-4}), batch size = 32

4. Results

To analyze and compare the overall performance of LLMs, experiments were conducted across five datasets with forecasting horizons of 1, 3, 6, 36, 72, 144, and 432 hours. The forecast errors of LLMs, along with comparisons with other models, are presented in Tables 3 and 4 across the five datasets.

Table 3
MAE of different models on five datasets

Data set	Models	Forecasting horizon (step)						
		1	**3**	**6**	**36**	**72**	**144**	**432**
Aktobe	ARIMA	0.3692 ±0.012	0.6565 ±0.015	1.1874 ±0.021	2.1598 ±0.035	2.6912 ±0.041	3.1485 ±0.045	3.4411 ±0.051
	SegRNN	0.0917 ±0.005	0.2008 ±0.007	0.3514 ±0.009	0.6635 ±0.012	0.7951 ±0.015	0.7892 ±0.014	0.8341 ±0.018
	PatchTST	0.1161 ±0.006	0.1975 ±0.008	0.3218 ±0.011	0.6622 ±0.014	0.7431 ±0.012	0.7825 ±0.015	0.8182 ±0.017
	BART	0.0931 ±0.004	0.1985 ±0.006	0.3223 ±0.008	0.6605 ±0.011	0.7361 ±0.010	0.7725 ±0.011	0.7963 ±0.012
	BERT	0.0712 ±0.003	0.1528 ±0.005	0.3142 ±0.007	0.6614 ±0.009	0.7295 ±0.008	0.7704 ±0.010	0.7955 ±0.011

Continuation of the table

Data set	Models	Forecasting horizon (step)						
		1	**3**	**6**	**36**	**72**	**144**	**432**
Aktau	ARIMA	0.3533 ±0.011	0.6701 ±0.016	1.1805 ±0.020	2.1612 ±0.032	2.6645 ±0.038	3.1388 ±0.042	3.4315 ±0.048
	SegRNN	0.0911 ±0.005	0.2025 ±0.008	0.3541 ±0.010	0.6455 ±0.011	0.6885 ±0.014	0.7824 ±0.016	0.8361 ±0.019
	PatchTST	0.0998 ±0.004	0.1981 ±0.007	0.3238 ±0.009	0.6295 ±0.012	0.7152 ±0.013	0.7845 ±0.015	0.8126 ±0.018
	BART	0.0931 ±0.003	0.1985 ±0.005	0.3218 ±0.007	0.6292 ±0.010	0.6815 ±0.011	0.7802 ±0.012	0.8118 ±0.014
	BERT	0.0901 ±0.003	0.1648 ±0.005	0.3095 ±0.006	0.6195 ±0.009	0.6766 ±0.010	0.7775 ±0.011	0.7996 ±0.013
Kokshetau	ARIMA	0.3942 ±0.013	0.6631 ±0.017	1.1965 ±0.022	2.2155 ±0.036	2.6892 ±0.040	3.2175 ±0.047	3.4355 ±0.052
	SegRNN	0.1292 ±0.006	0.1818 ±0.007	0.3625 ±0.011	0.6432 ±0.013	0.6908 ±0.015	0.7925 ±0.017	0.8105 ±0.018
	PatchTST	0.1155 ±0.005	0.1738 ±0.006	0.3218 ±0.009	0.6295 ±0.011	0.6912 ±0.013	0.7442 ±0.015	0.7785 ±0.016
	BART	0.1135 ±0.004	0.1848 ±0.006	0.3202 ±0.008	0.6265 ±0.010	0.6915 ±0.011	0.7365 ±0.012	0.7792 ±0.014
	BERT	0.1042 ±0.004	0.1752 ±0.005	0.2965 ±0.007	0.6192 ±0.009	0.6885 ±0.010	0.7342 ±0.011	0.7735 ±0.012
Taldykorgan	ARIMA	0.3521 ±0.011	0.6138 ±0.015	1.1685 ±0.021	1.9855 ±0.031	2.4582 ±0.037	3.0265 ±0.042	3.2562 ±0.048
	SegRNN	0.0812 ±0.004	0.1992 ±0.007	0.3508 ±0.009	0.6612 ±0.012	0.6822 ±0.014	0.7855 ±0.016	0.8235 ±0.019
	PatchTST	0.0858 ±0.004	0.1768 ±0.006	0.3355 ±0.009	0.6262 ±0.011	0.6805 ±0.013	0.7872 ±0.015	0.8208 ±0.018
	BART	0.0818 ±0.003	0.1778 ±0.005	0.3222 ±0.008	0.6355 ±0.010	0.6812 ±0.011	0.7865 ±0.013	0.8122 ±0.014
	BERT	0.0782 ±0.003	0.1695 ±0.005	0.3205 ±0.007	0.6265 ±0.009	0.6775 ±0.010	0.7725 ±0.012	0.8118 ±0.013
Uralsk	ARIMA	0.3592 ±0.012	0.6542 ±0.016	1.1722 ±0.021	2.1485 ±0.035	2.6765 ±0.040	3.1254 ±0.045	3.4585 ±0.051
	SegRNN	0.0905 ±0.005	0.1988 ±0.007	0.3452 ±0.010	0.6565 ±0.012	0.6822 ±0.014	0.7765 ±0.016	0.8175 ±0.018
	PatchTST	0.1188 ±0.005	0.2085 ±0.008	0.3225 ±0.009	0.6555 ±0.012	0.6775 ±0.013	0.7812 ±0.015	0.8021 ±0.017
	BART	0.0894 ±0.004	0.1978 ±0.006	0.3215 ±0.008	0.6512 ±0.010	0.6755 ±0.011	0.7715 ±0.012	0.7978 ±0.014
	BERT	0.0895 ±0.003	0.1922 ±0.005	0.3195 ±0.007	0.6562 ±0.009	0.6795 ±0.010	0.7445 ±0.011	0.8005 ±0.013

Table 4
RMSE of different models on five datasets

Data set	Models	Forecasting horizon (step)						
		1	**3**	**6**	**36**	**72**	**144**	**432**
Aktobe	ARIMA	0.4576 ±0.014	0.8318 ±0.020	1.4725 ±0.028	2.7391 ±0.042	3.3362 ±0.049	3.9125 ±0.055	4.2685 ±0.062
	SegRNN	0.1162 ±0.006	0.2541 ±0.009	0.4362 ±0.012	0.8255 ±0.016	1.0092 ±0.019	0.9785 ±0.018	1.0592 ±0.021
	PatchTST	0.1468 ±0.007	0.2448 ±0.010	0.4091 ±0.013	0.8382 ±0.018	0.9245 ±0.017	0.9935 ±0.020	1.0178 ±0.022
	BART	0.1154 ±0.005	0.2525 ±0.008	0.4002 ±0.010	0.8385 ±0.015	0.9128 ±0.014	0.9778 ±0.016	1.0108 ±0.018
	BERT	0.0892 ±0.004	0.1948 ±0.006	0.3915 ±0.009	0.8371 ±0.013	0.9252 ±0.012	0.9775 ±0.014	1.0092 ±0.016
Aktau	ARIMA	0.4488 ±0.013	0.8285 ±0.019	1.4635 ±0.026	2.7395 ±0.041	3.3698 ±0.048	3.8925 ±0.054	4.3452 ±0.060
	SegRNN	0.1152	0.2512	0.4505	0.8188	0.8535	0.9898	1.0575

Continuation of the table

Data set	Models	Forecasting horizon (step)						
		1	3	6	36	72	144	432
		±0.006	±0.009	±0.013	±0.015	±0.017	±0.020	±0.022
	PatchTST	0.1248 ±0.005	0.2472 ±0.008	0.4115 ±0.011	0.7805 ±0.014	0.9075 ±0.016	0.9728 ±0.018	1.0282 ±0.020
	BART	0.1176 ±0.004	0.2528 ±0.007	0.4012 ±0.009	0.7962 ±0.012	0.8458 ±0.014	0.9902 ±0.016	1.0065 ±0.018
	BERT	0.1118 ±0.004	0.2061 ±0.006	0.3855 ±0.008	0.7842 ±0.011	0.8562 ±0.013	0.9672 ±0.015	0.9941 ±0.017
Kokshetau	ARIMA	0.4912 ±0.015	0.8222 ±0.020	1.4885 ±0.028	2.8085 ±0.043	3.3358 ±0.050	4.0712 ±0.058	4.2625 ±0.063
	SegRNN	0.1635 ±0.007	0.2272 ±0.009	0.4498 ±0.013	0.8005 ±0.016	0.8765 ±0.018	1.0022 ±0.021	1.0275 ±0.022
	PatchTST	0.1432 ±0.006	0.2172 ±0.008	0.3992 ±0.011	0.7835 ±0.014	0.8575 ±0.016	0.9255 ±0.018	0.9855 ±0.020
	BART	0.1451 ±0.005	0.2338 ±0.007	0.3972 ±0.010	0.7772 ±0.012	0.8582 ±0.014	0.9342 ±0.016	0.9662 ±0.018
	BERT	0.1332 ±0.004	0.2221 ±0.006	0.3751 ±0.009	0.7708 ±0.011	0.8561 ±0.013	0.9288 ±0.015	0.9812 ±0.017
Taldykorgan	ARIMA	0.4452 ±0.013	0.7785 ±0.018	1.4495 ±0.026	2.4668 ±0.038	3.0495 ±0.045	3.7592 ±0.052	4.0412 ±0.058
	SegRNN	0.1021 ±0.005	0.2522 ±0.009	0.4455 ±0.012	0.8225 ±0.016	0.8632 ±0.018	0.9965 ±0.021	1.0425 ±0.023
	PatchTST	0.1075 ±0.005	0.2252 ±0.008	0.4245 ±0.011	0.7922 ±0.014	0.8445 ±0.016	0.9785 ±0.019	1.0205 ±0.021
	BART	0.1038 ±0.004	0.2222 ±0.007	0.3998 ±0.010	0.8062 ±0.013	0.8622 ±0.015	0.9951 ±0.018	1.0282 ±0.020
	BERT	0.0972 ±0.003	0.2162 ±0.006	0.3991 ±0.009	0.7925 ±0.011	0.8592 ±0.014	0.9782 ±0.016	1.0295 ±0.018
Uralsk	ARIMA	0.4455 ±0.013	0.8135 ±0.019	1.4835 ±0.027	2.7242 ±0.041	3.3875 ±0.049	4.0265 ±0.056	4.2915 ±0.061
	SegRNN	0.1148 ±0.005	0.2488 ±0.009	0.4281 ±0.012	0.8175 ±0.016	0.8635 ±0.018	0.9855 ±0.020	1.0172 ±0.022
	PatchTST	0.1488 ±0.006	0.2642 ±0.010	0.4005 ±0.011	0.8298 ±0.015	0.8435 ±0.016	0.9688 ±0.019	0.9955 ±0.021
	BART	0.1112 ±0.004	0.2475 ±0.008	0.4092 ±0.010	0.8082 ±0.013	0.8551 ±0.014	0.9785 ±0.017	1.0122 ±0.019
	BERT	0.1145 ±0.004	0.2435 ±0.007	0.4045 ±0.010	0.8335 ±0.014	0.8605 ±0.015	0.9238 ±0.016	1.0155 ±0.018

Experimental results on five wind speed datasets show a large difference in performance between the models. Transformer-based baselines and the proposed frozen pre-trained Transformer backbone models, such as PatchTST, BART, and BERT, significantly outperform the classical ARIMA model. This difference is measured using two metrics, MAE and RMSE, over forecasting horizons ranging from 1 to 432 steps. In addition, the resulting errors also increase as the forecasting horizon increases. This is a common feature of long-term forecasting, reflecting increased uncertainty and the accumulation of errors over time.

The ARIMA model has the highest error in most cases. With long forecasting steps (144-432), the RMSE error increases sharply, reaching from 3.9 to 4.4. This shows the limitation of the linear model in describing the nonlinear characteristics and fluctuations of wind.

Meanwhile, the SegRNN and PatchTST models significantly improve the forecasting results at short time steps (1-72) with MAE ranging from 0.08 to 0.354 and RMSE ranging from 0.1 to 1. This shows that the latter models better capture the temporal relationship than ARIMA.

For BART and BERT, these models achieve the best results on most datasets and forecasting steps. In particular, with a long forecasting step of 432, the MAE and RMSE of BERT are 50-70% lower than those of ARIMA. These results indicate that frozen pre-trained Transformer backbones can achieve better accuracy in long-term forecasting under the present experimental setting.

Geographically, regions exposed to strong continental steppe winds (such as Aktobe) and coastal areas (such as Aktau) exhibit higher errors at long forecasting steps due to stronger, less predictable wind fluctuations.

Figures 3 and 4 show the MAE and RMSE plots of the models on the five datasets, respectively. MAE and RMSE increase with the forecasting step. However, the increase in the proposed BERT/BART-based models is slower and more stable than that of other models, suggesting that the frozen pre-trained Transformer blocks provide effective long-range sequence modeling in this forecasting task.

To assess whether the performance differences were statistically significant, the Diebold-Mariano (DM) test was conducted to compare the proposed BERT and BART models against the strongest baseline, PatchTST. The complete DM statistics and corresponding p-values for all forecasting horizons are provided in the Appendix A2. Overall, the DM test results support the superiority of the proposed models across most forecasting horizons, with stronger statistical evidence observed particularly at the 144-step and 432-step horizons.

While models achieve high accuracy using standard metrics (MSE, MAE), these metrics treat all errors equally. In reality, wind power generation is proportional to the cube of wind speed $P \propto v^3$. Therefore, forecasting errors at high wind speeds impact grid stability much more severely than those at low speeds. Future research should integrate a power-weighted loss function during LLM fine-tuning to better capture extreme wind dynamics and align with real-world operational requirements.

Figure 3
Comparison of the MAE of different models across five cities in Kazakhstan

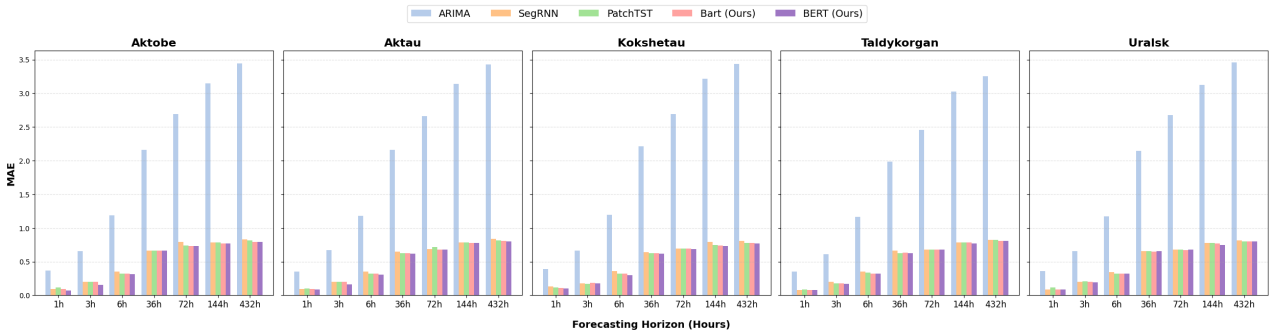
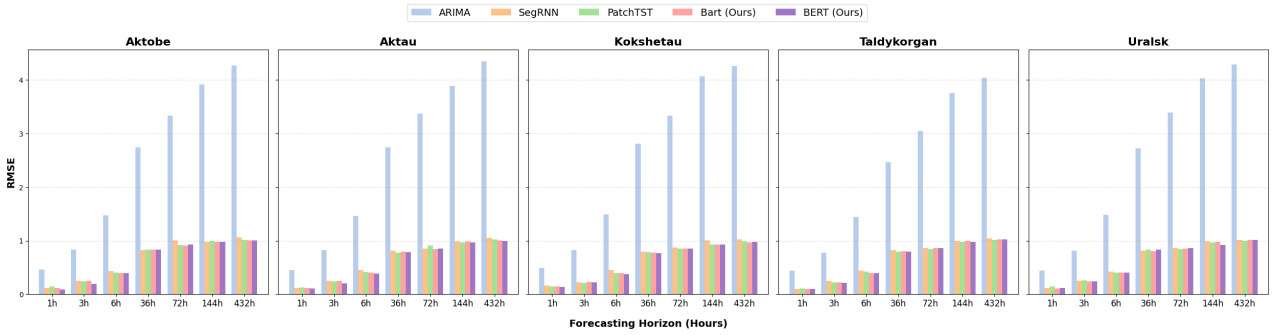


Figure 4
Comparison of the RMSE of different models across five cities in Kazakhstan



5. Conclusion

This study proposed a wind speed forecasting framework based on frozen pre-trained Transformer backbones (BERT and BART) to improve forecasting accuracy across different climatic and topographic regions of Kazakhstan. Five different datasets and seven forecasting horizons were used to evaluate the performance of the proposed framework. Under the present experimental setup, the proposed BERT- and BART-based models achieved lower MAE and RMSE than the selected statistical and deep learning baselines, while maintaining relatively stable performance at long forecasting horizons. The results suggest that frozen pre-trained Transformer backbones, when combined with lightweight adaptation, can effectively model nonlinear, non-stationary, and long-range temporal dependencies in wind-speed data. Overall, this study supports the potential of pre-trained Transformer backbones as promising tools for wind-speed forecasting and provides an application-oriented reference for planning and operation in wind power systems.

Funding

The research has been funded by the Science Committee of the Ministry of Science and Higher Education of the Republic of Kazakhstan (Grant No. AP26198931).

Conflicts of Interest

The authors declare no conflict of interest.

Appendix

Table A1

Summary aggregated across the five datasets of Diebold-Mariano test results for BERT/BART versus PatchTST across forecasting horizons

Horizon	BERT vs PatchTST	BART vs PatchTST
1	DM = 1.72, (p=0.089)	DM = 1.81, (p=0.074)
3	DM = 1.96, (p=0.054)	DM = 2.04, (p=0.046)
6	DM = 2.11, (p=0.039)	DM = 2.18, (p=0.033)
36	DM = 2.28, (p=0.027)	DM = 2.35, (p=0.022)
72	DM = 2.41, (p=0.021)	DM = 2.53, (p=0.016)
144	DM = 2.86, (p=0.008)	DM = 2.97, (p=0.006)
432	DM = 3.12, (p=0.004)	DM = 3.26, (p=0.002)

Author Contributions

Conceptualization, T.T.C. and A.B.A.; Methodology, T.T.C.; Software, T.T.C.; Validation, T.T.C., M.S. and K.T.; Formal Analysis, T.T.C.; Investigation, T.T.C., M.S. and M.A.S.; Resources, A.B.A.; Data Curation, T.T.C.; Writing – Original Draft Preparation, T.T.C.; Writing – Review & Editing, T.T.C., M.A.S. and A.B.A.; Visualization, T.T.C.; Supervision, A.B.A.; Project Administration, A.B.A.; Funding Acquisition, A.B.A.

References

1. T. T. Chau, A. B. Alhassan, M. Shaltayev, K. Talapiden, M. A. Shehu, and T. D. Do, “Assessment of wind energy potential using hybrid adaptive bandwidth kernel density technique: A case study of major cities in Kazakhstan,” *Energy Conversion and Management: X*, vol. 29, p. 101439, Jan. 2026, doi: 10.1016/j.ecmx.2025.101439.

2. U. Alparslan, “Green energy corridors for Central Asia and the Caucasus”. Accessed: Feb. 22nd, 2026. [Online]. Available: https://ember-energy.org/app/uploads/2024/11/Report-Green-energy-corridors-for-Central-Asia-and-the-Caucasus.pdf

3. IEA, “Kazakhstan 2022”, Accessed: Oct. 09, 2025. [Online]. Available: https://iea.blob.core.windows.net/assets/fc84229e-6014-4400-a963-bccea29e0387/Kazakhstan 2022.pdf

4. T. T. Chau, T. N. Nguyen, and T. D. Do, “Assessing wind energy exploitation potential in several regions of Viet Nam using Kernel density estimation model,” *CTU Journal of Innovation and Sustainable Development*, vol. 16, pp. 25–34, 2024, doi: 10.22144/ctujoisd.2024.319.

5. T. T. Chau, T. T. H. Nguyen, L. Nguyen, and T. D. Do, “Wind Speed Probability Distribution Based on Adaptive Bandwidth Kernel Density Estimation Model for Wind Farm Application,” *Wind Energy*, vol. 28, no. 2, p. e2970, Feb. 2025, doi: 10.1002/WE.2970.

6. A. M. Khan, A. B. Alhassan, A. Kolumbetov, A. Haruna, V. Gali, N. G. M. Thao, and T. D. Do, “Optimizing demand-side energy management for stand-alone wind-solar microgrids in rural settlements: A case study for nomadic Yurt in Kazakhstan,” *Energy Conversion and Management: X*, vol. 30, p. 101587, May 2026, doi: 10.1016/j.ecmx.2026.101587.

7. T. T. Chau, H. T. Nguyen, and T. D. Do, “Online Nonlinear Quadratic Tracking Control Accompanied by Selective Current Harmonic Compensator for MPPT of PMSG Drives,” *IEEE Journal of Emerging and Selected Topics in Power Electronics*, 2026, doi: 10.1109/JESTPE.2026.3656176.

8. T. D. Do, H. T. Nguyen, A. S. Al-Sumaiti, K. Al Hosani, and D. D. Nguyen, “Nonlinear Quadratic Regulator Design With a Hybrid Switching Scheme for Wind Turbine Generators,” *IEEE Trans Syst Man Cybern Syst*, vol. 53, no. 9, pp. 5525–5535, Sep. 2023, doi: 10.1109/TSMC.2023.3272222.

9. T. T. Chau, T. N. Nguyen, L. Nguyen, A. B. Alhassan, and T. D. Do, “A maximum power point tracking control for wind energy conversion systems using regularized data-enabled predictive control,” *Engineering Applications of Artificial Intelligence*, vol. 170, p. 114005, Apr. 2026, doi: 10.1016/j.engappai.2026.114005.

10. M. A. Shehu, K. Talapiden, T. T. Chau, A. Haruna, M. Aly, V. Gali, T. D. Do, and A. B. Alhassan, “Control and energy management of standalone microgrids in remote areas: A review of recent advances, challenges, and opportunities for future research,” *Engineering Science and Technology, an International Journal*, vol. 75, p. 102288, Mar. 2026, doi: 10.1016/j.jestch.2026.102288.

11. G. Zhu, W. Jia, Z. Xing, L. Xiang, A. Hu, and R. Hao, “CMLLM: A novel cross-modal large language model for wind power forecasting,” *Energy Convers Manag*, vol. 330, p. 119673, Apr. 2025, doi: 10.1016/J.ENCONMAN.2025.119673.

12. H. R. Alsamamra, S. Salah, and J. H. Shoqeir, “Performance analysis of ARIMA Model for wind speed forecasting in Jerusalem, Palestine,” *Energy Exploration & Exploitation*, vol. 42, no. 5, pp. 1727–1746, Sep. 2024, doi: 10.1177/01445987241248201.

13. L. Xiaolei, L. Zi, and F. Ziming, “Short-term offshore wind speed forecast by seasonal ARIMA – A comparison against GRU and LSTM,” *Energy (Oxf.)*, vol. 227, no. 120492 Jul 2021, doi: 10.1016/j.energy.2021.120492.

14. H. Liu, H. qi Tian, and Y. fei Li, “Comparison of two new ARIMA-ANN and ARIMA-Kalman hybrid methods for wind speed prediction,” *Appl Energy*, vol. 98, pp. 415–424, Oct. 2012, doi: 10.1016/J.APENERGY.2012.04.001.

15. G. E. P. Box, G. M. Jenkins, G. C. Reinsel, and G. M. Ljung, “JOURNAL OF TIME SERIES ANALYSIS BOOK REVIEW TIME SERIES ANALYSIS: FORECASTING AND CONTROL, 5TH EDITION, by,” *J. Time. Ser. Anal*, vol. 37, pp. 709–711, 2016, Accessed: Oct. 09, 2025. [Online]. Available: https://www.wiley.com/en-us/Time+Series+Analysis%3A+Forecasting+and+Control%2C+5th+Edition-p-9781118675021

16. J. Duan *et al.*, “A combined short-term wind speed forecasting model based on CNN–RNN and linear regression optimization considering error,” *Renew Energy*, vol. 200, pp. 788–808, Nov. 2022, doi: 10.1016/J.RENENE.2022.09.114.

17. J. Zheng and J. Wang, “Short-term wind speed forecasting based on recurrent neural networks and Levy crystal structure algorithm,” *Energy*, vol. 293, p. 130580, Apr. 2024, doi: 10.1016/J.ENERGY.2024.130580.

18. X. Ai, S. Li, and H. Xu, “Short-term wind speed forecasting based on two-stage preprocessing method, sparrow search algorithm and long short-term memory neural network,” *Energy Reports*, vol. 8, pp. 14997–15010, Nov. 2022, doi: 10.1016/J.EGYR.2022.11.051.

19. K. Cho et al., "Learning phrase representations using RNN encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.

20. Z. Fu *et al.*, “An Informer-BiGRU-temporal attention multi-step wind speed prediction model based on spatial-temporal dimension denoising and combined VMD decomposition,” *Energy*, vol. 326, p. 136265, Jul. 2025, doi: 10.1016/J.ENERGY.2025.136265.

21. G. Ban, Y. Chen, Z. Xiong, Y. Zhuo, and K. Huang, “The univariate model for long-term wind speed forecasting based on wavelet soft threshold denoising and improved Autoformer,” *Energy*, vol. 290, p. 130225, Mar. 2024, doi: 10.1016/J.ENERGY.2023.130225.

22. T. Zhou, Z. Ma, Q. Wen, X. Wang, L. Sun, and R. Jin, “FEDformer: Frequency Enhanced Decomposed Transformer for Long-term Series Forecasting,” *Proc Mach Learn Res*, vol. 162, pp. 27268–27286, Jan. 2022, Accessed: Oct. 09, 2025. [Online]. Available: https://arxiv.org/pdf/2201.12740

23. J. Devlin, M. W. Chang, K. Lee, and K. Toutanova, “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding,” *NAACL HLT 2019 – 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies – Proceedings of the Conference*, vol. 1, pp. 4171–4186, Oct. 2018, Accessed: Oct. 09, 2025. [Online]. Available: https://arxiv.org/pdf/1810.04805

24. M. Lewis *et al.*, “BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension,” *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pp. 7871–7880, Oct. 2019, doi: 10.18653/v1/2020.acl-main.703.

25. A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, “Improving Language Understanding by Generative Pre-Training”, Accessed: Oct. 09, 2025. [Online]. Available: https://cdn.openai.com/research-covers/language-unsupervised/language_understanding_paper.pdf

26. H. Touvron *et al.*, "LLaMA: Open and Efficient Foundation Language Models," *arXiv preprint arXiv:2302.13971*, 2023. [Online]. Available: https://arxiv.org/abs/2302.13971

27. M. Jin *et al.*, “Time-LLM: Time Series Forecasting by Reprogramming Large Language Models,” *12th International Conference on Learning Representations, ICLR 2024*, Oct. 2023, Accessed: Oct. 09, 2025. [Online]. Available: https://arxiv.org/pdf/2310.01728

Information about the authors:

Tin Trung Chau received the B.S. degree in Automation and Control Engineering Technology and the M.S. degree in Electrical Engineering from the Vinh Long University of Technology Education, Vietnam, in 2020 and 2022, respectively. He is a Ph.D. student in Robotics at the School of Engineering and Digital Sciences, Nazarbayev University, Kazakhstan. His research interests include robotics, renewable energy conversion systems, and machine learning (ORCID iD: 0009-0009-7831-8864).

Miras Shaltayev is a BSc student in Computer Science at the School of Engineering and Digital Sciences at Nazarbayev University. Kazakhstan. His research interests include machine learning, deep learning and renewable energy (ORCID iD: 0009-0008-4457-3763).

Kulash Talapiden received her Bachelor's degree (Hons.) in Technological Machinery and Equipment from Kazakh Agrotechnical University in 2020, and her M.S. degree in Robotics from Nazarbayev University. Her research interests include

control systems, optimization techniques for control system design and analysis, feedback control in motor drives, and the control of autonomous systems (ORCID iD: 0000-0002-1757-7311).

Muhammad Auwal Shehu received the B.Eng. degree in electrical engineering from Bayero University, Kano, Nigeria, in 2012 and the M.Eng. degree in mechatronics and automatic control from the University of Technology Malaysia in 2016. His primary research interests include the design of nonlinear control systems and state estimators for underactuated mechatronic systems (ORCID iD: 0000-0002-2158-7619).

Ahmad Bala Alhassan received a Ph.D. degree in Mechanical Engineering from Xi'an Jiaotong University, China, in 2022. He is currently a Postdoctoral Scholar in the Department of Robotics and Mechatronics at Nazarbayev University, Kazakhstan. His current research interests include modeling, simulation, and control of mechatronic systems. He has also served as a reviewer for many refereed journals, including IEEE Access, ISA Transactions, and Robotics and Autonomous Systems. He has been a Member of IEEE, including IEEE Young Professionals, IEEE Control Systems Society, and IEEE Industrial Electronics Society (ORCID iD: 0000-0002-0179-7686).

Submission received: 19 January, 2026
Revised: 16 March, 2026
Accepted: 16 March, 2026

IRSTI 28.23.25; 81.93.29 https://doi.org/10.26577/jpcsit4220267

Aerna Aheti , Hadi Mukhtar*

Al-Farabi Kazakh National University, Almaty, Kazakhstan
*e-mail: hadimukhtar102@gmail.com

APPLICATION OF MACHINE LEARNING METHODS FOR PHISHING ATTACK DETECTION: A COMPARATIVE ANALYSIS OF MODELS AND THEIR EFFECTIVENESS

Abstract. Despite significant progress in combating, phishing continues to be one of the top cyber-security risks, capitalizing on both technological gaps and human behavior. In this work, we propose a machine learning (ML) framework for creating phishing URL detector based on the LegitPhish labeled dataset that consists of 101,219 URLs and 17 engineered features for each URL. There is a particular focus on pre-modelling data diagnostics to resolve any data leakage, shortcuts and highly correlated variables that can drive the model results and cause artificial over-optimism. Six classification models were tested such as Logistic Regression (LR), Support Vector Machine (SVM), Random Forest, XGBoost, Multi-Layer Perceptron (MLP) and CNN-LSTM architecture. Seven informative features were selected for model development: After discarding two leakage-prone features has_ip_address, https_flag and eight redundant features. Experimental results revealed that the overall performance of the two models, Random Forest and XGBoost, were the best with an F1-score of 0.952 and an ROC-AUC value of 0.992 respectively. The precision score for the Random Forest model was 0.928 and the Recall score was 0.978, which provides a good balance between the detection of phishing and control of false positives. Despite the complexity of the CNN-LSTM model, it did not perform better than the ensemble-based models, while the MLP model proved competitive. The results confirm the significance of careful feature diagnostics and leakage prevention and show that well-tuned ensemble classifiers are able to deliver accurate, efficient, and viable results for the phishing URL detection task.

Keywords: phishing detection, machine learning, random forest, cybersecurity, ensemble methods, data leakage.

1. Introduction

Phishing stands as a type of cyberattack that changes constantly and stays very strong when phishing targets the most vulnerable part of the security system, which is the person [1]. Due to 1,350,037 attacks occurring, the results suggest that phishing attacks reached a peak in 2022. A record number of 1,350,037 attacks was recorded during 2022, which is nearly double the 611,877 attacks seen in 2021 [2]. Monetary organizations, social interaction sites, and digital storage providers were affected the most. The attack process has evolved over three decades. The first wave (1995–2005) consisted of crude, mass-emailing campaigns with spelling errors and generic greetings [3]. The second generation (2005–2015) introduced technically advanced spear-phishing attacks using targeted social engineering and customized websites [4].

The third generation (2015–present) presents new challenges, marked by the offensive use of artificial intelligence. Attackers now employ ML models to automate highly targeted lures, deepfakes for believable synthetic media, and polymorphic infrastructure that evades signature-based detection [5]. Traditional defenses have proven inadequate. Blacklist-based filtering suffers from the "zero-day" or "first victim" problem, failing to protect against unseen threats [6]. Heuristic systems generate excessive false positives and are easily bypassed by adversaries who reorder URL components or adopt novel obfuscation techniques [7].

Recent surveys have comprehensively reviewed the landscape of phishing detection techniques [8], [9]. Machine learning offers a proactive alternative by learning complex, nonlinear relationships from data, enabling adaptation to previously unseen threats [10]. Numerous studies have investigated ML and deep learning models for phishing classification, including attention-based bidirectional LSTM networks [11] and transformer-based architectures [12]. However, a recurring problem is data leakage: features that are nearly perfectly correlated with the target but lack real-

world validity [13]. Models trained on such features perform well in the lab but fail in production.

This paper conducts a methodologically rigorous comparison of ML models on a level playing field, free from data leakage distortions.

We test the hypothesis that after proper data diagnostics, well-tuned ensemble tree-based models will match or outperform more complex deep learning architectures on structured tabular data. We perform comprehensive diagnostics on the LegitPhish dataset [14], [15], eliminate problematic features, and evaluate six models spanning classical, ensemble, and deep learning families.

2. Materials and methods

2.1 The LegitPhish Dataset and Pre-Modeling Data Diagnostics

Counted labeled URLs reach 101,219 within the LegitPhish dataset [14] [15] with 17 engineered features that show lexical (e.g., url length, url entropy), structural (e.g., dot count, subdomain count), and semantic (e.g., tld popularity, suspicious file extension) qualities. Phishing URLs are assigned the class 0 and legitimate URLs are assigned the class 1 from the original LegitPhish dataset. During the preprocessing step, the labels were also switched to make the phishing URL a "class 1" and the legitimate URL a "class 0", which is consistent with the binary classification conventions used in machine learning. This transformed encoding is used for all analyses, correlation calculations, and evaluation metrics and model outputs reported in this study.

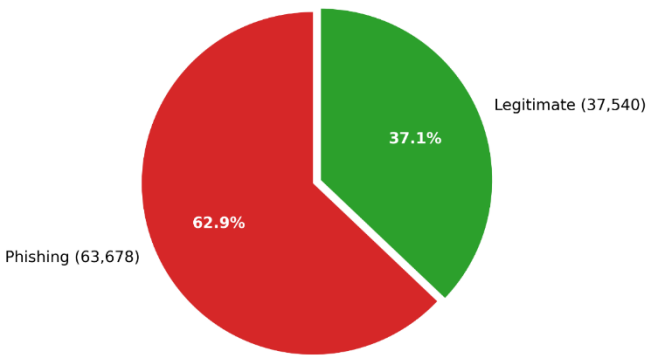
Class distribution contains 63,678 phishing instances (63%) and 37,540 legitimate instances (37%), which makes the LegitPhish dataset good for testing binary classification when the balance is moderate. Because the class distribution contains 63,678 phishing instances (63%) and 37,540 legitimate instances (37%), evaluation of binary classification is performed under moderate imbalance.

The dataset contains 63,678 phishing URLs the class distribution is shown in Figure 1. (63%) and 37,540 legitimate URLs (37%).

Table 1
Sample entries from the LegitPhish dataset

URL	url length	dot count	https flag	path length	tld length	Class Label
https://keraekken-loagginnusa.godaddysites.com/	47	2	1	1	3	0
https://metamsk01lgiix.godaddysites.com/	40	2	1	1	3	0
http://myglobaltech.in/	23	1	0	1	2	0
http://djtool-for-spotify.com/	30	1	0	1	3	0
https://scearmcoommunnlty.com/invent/freind/get	47	1	1	18	3	0

Figure 1
Class distribution in the LegitPhish dataset



Before any model training, we performed three diagnostic analyses:

Perfect Predictor Identification: Cross-tabulation revealed that has ip address = 1 occurred only in phishing URLs. Every URL containing an IP address was phishing. However, many phishing URLs do not contain IP addresses. This is a dataset-specific artifact (one-directional indicator), not a generalizable perfect predictor. This feature was removed.

High-Correlation Analysis: A correlation matrix showed https flag had an absolute correlation of |r| = 0.929 with the target near-perfect. In this dataset, HTTPS presence almost always indicated a legitimate URL, which oversimplifies real-world conditions where phishers increasingly use HTTPS [16]. This feature was removed.

Redundancy Test: Characteristics with inter-correlation |r| > 0.7 like token count, number of digits, and percentage numeric chars were viewed as repetitive.

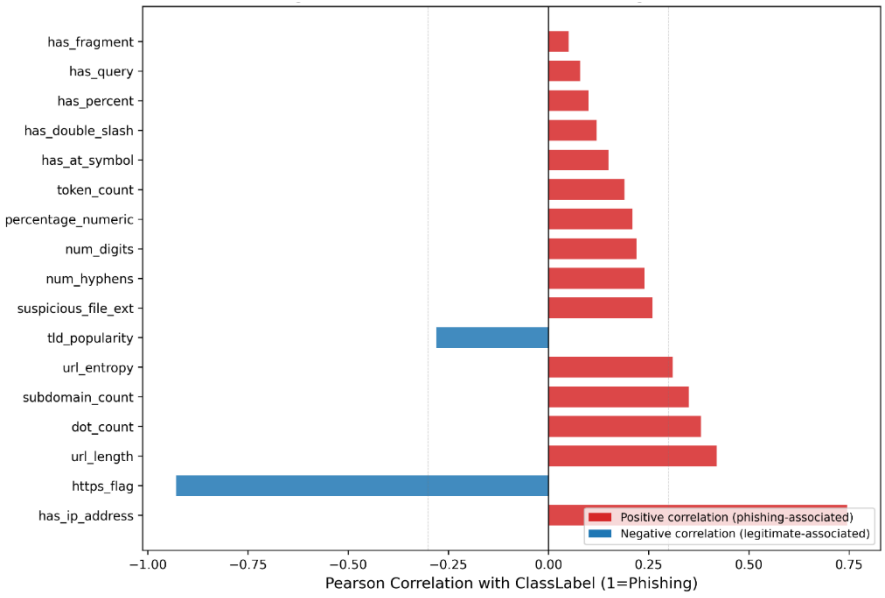
To get rid of variables that are very redundant and shortcut features that are specific to the dataset, feature removal was performed. Seven of these variables were informative and showed reasonable correlation and improved model interpretability, and they were kept in the model's final feature set.

Final feature count: 17 original features − 2 leakage features − 8 redundant features = 7 retained features.

Table 2
Feature Selection Process

Feature	Action	Reason
has ip address	Removed	Dataset-specific shortcut feature; all URLs containing IP addresses were phishing
https flag	Removed	Near-perfect correlation with target (
token count	Removed	Highly correlated with url length
number of digits	Removed	Highly correlated with percentage numeric chars
percentage numeric chars	Removed	Redundant information
special char count	Removed	High inter-feature correlation
path length	Removed	Redundant with URL complexity metrics
hostname length	Removed	Highly correlated with url length
digit to letter ratio	Removed	Redundant feature
suspicious word count	Removed	High correlation with lexical complexity measures
Remaining 7 features	Retained	Correlation below threshold and meaningful predictive contribution

Figure 2
Horizontal bar chart showing Pearson correlation of each original feature with the target variable



Data in the figure shows that https flag has strong negative correlation (r = -0.929) and has ip address has positive correlation (r = +0.745) with the target (Phishing=1). A strong negative relationship is displayed by the https flag variable while the has ip address factor connects to the goal because the numbers are high. Results show that these specific markers provide clear signals for identifying fraudulent sites.

Shows all IP-containing URLs are phishing (4,812 instances), but many phishing URLs (58,866) lack IPs a one-directional artifact, not a perfect predictor.

All absolute correlations with the target are below 0.7, confirming successful feature selection.

Recent work has demonstrated the effectiveness of deep learning architectures such as GAN-CNN-LSTM for phishing detection [17], while comparative studies have evaluated multiple classification algorithms on various phishing datasets [18]. The dataset was first split as training and testing partitions by stratified sampling for avoiding information leakage. All diagnostics, correlation analysis, feature selection, scaling and hyperparameter tuning were done only on the training data. The subset of features and the parameters used for preprocessing were then fixed and applied to the test set which was held out. This method prevented any information from the test set from affecting model development.

Figure 3
Confusion matrix: has ip address vs. target

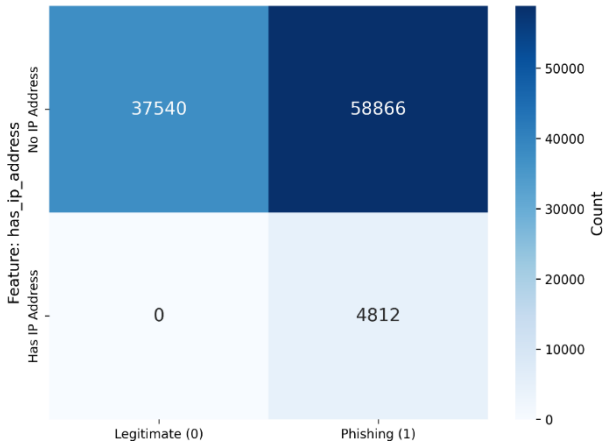


Figure 4
Correlation heatmap of the final 7 retained features

2.2 Model development

After the feature selection process described in Section 2.1, the final dataset with 7 optimized features was divided into the training and testing sets by 70/30 stratified random sampling maintaining the same class ratio (63%/37%) for phishing/legitimate classes. To standardise features, StandardScaler (zero mean, unit variance) was used — this is important for distance based methods (SVM) and gradient decent methods (neural networks) but not for tree based methods (Random Forest, XGBoost).

Implementation code of all the models were written in Python 3.10 with scikit-learn version 1.2.0, XGBoost version 1.7.0 and TensorFlow version 2.12.0. Hyperparameter tuning was done on the training set only, which was done by random search with 5 fold cross validation. Cross validation was performed 5-fold stratified and resulted in the reporting of mean performance with standard deviation [22].

2.2.1 Logistic Regression (LR)

Logistic Regression (LR): is the baseline linear classifier because it is easy to understand and interpret [19]. It approximates the probability that a URL is phishing by applying sigmoid function:

$$P(Y = 1 \mid X) = \frac{1}{1 + e^{-(\beta_0 + \sum_{i=1}^{n} \beta_i x_i)}} \qquad (1)$$

Configuration: liblinear solver, L2 regularization with C=1.0, maximum number of iterations=1000

2.2.2 Support Vector Machine (SVM) with RBF Kernel

SVM computes the optimum separating hyperplane for the two classes, with the maximum possible margin. The RBF kernel can be used to make non-linear decision boundaries by mapping features into a higher dimensional space [18]:

$$f(x) = sign\left(\sum_{i=1}^{N} \alpha_i \, y_i K(x_i, x) + b\right) \qquad (2)$$

$$K(x_i, x) = e^{-\gamma \| x_i - x \|^2} \qquad (3)$$

Configuration: After hyperparameter tuning, optimal values were $C = 10.0$ (regularization) and $\gamma = 0.1$ (kernel width).

2.2.3 Random Forest (RF)

Random Forest is an ensemble of decision trees constructed by the bagging (bootstrap aggregating) and random feature selection at every split. Majority voting over all trees is used to make a final prediction [19].

This will minimise the variance but not affect the bias, hence this will make RF a good model that will not overfit. Configuration: 200 trees, no limit on the number of features per split, no limit on the tree depth, Bootstrap sampling is on. Out-of-bag (OOB) score was 0.961 which was comparable with the results of the cross-validation.

2.2.4 XGBoost (Extreme Gradient Boosting)

XGBoost constructs the trees one by one, each one attempting to rectify the errors of the ensemble tree that went before it, based on a regularized objective function [20]:

$$\text{Obj}^{(t)} = \\ = \sum_{i=1}^{n} l(y_i, \hat{y}_i^{(t-1)} + f_t(x_i)) + \Omega(f_t) \qquad (4)$$

where l is a differentiable convex loss function (binary logistic loss for classification) and $\Omega(f_t) = \gamma T + \frac{1}{2}\lambda \| w \|^2$ is the regularization term (T = number of leaves, w = leaf weights).

2.2.5 Multi-Layer Perceptron (MLP)

The MLP is a feedforward network with three hidden layers of decreasing size (64 → 32 → 16 neurons). Each dense layer is similar to the one in [21] which uses a linear transformation followed by an activation function called ReLU. We used generous regularization (50% dropout after every hidden layer, L2 kernel regularization (0.001) and batch normalization to stabilize the training process) to avoid overfitting.

$$h^{(l+1)} = \text{ReLU}(W^{(l)} h^{(l)} + b^{(l)}) \qquad (5)$$

The optimizer is Adam with learning rate of 0.001; batch size = 64; training is done with 100 epochs and early stopping (patience = 20). Training validation loss curves are shown in Figure 7.

2.2.6 CNN-LSTM Hybrid

The CNN-LSTM hybrid model consists of a 1D convolutional layer (32 filters, 2 kernel size)

followed by an LSTM layer (24 units). This is the type of architecture that is normally used for sequential data such as time series and textual.

We intentionally made it to see if a same signifying sample of unordered and independent features could be mined by such a model, given that this is a fundamentally misaligned inductive bias [21]. This added complexity is expected to give no advantage over tree-based methods (which is borne out in Section 3). Results: Note that the same optimizer, batch size, epochs, and regularization are used as for the MLP but dropout is set, after each layer, to 0.5.

2.2.7 Loss function

For all neural network models the loss function minimized during the learning phase was binary cross-entropy (log loss):

$$L_{CE} == -\frac{1}{N}\sum_{i=1}^{N}[y_i \log(p_i) + (1-y_i)\log(1-p_i)] \quad (6)$$

y i ∈ {0,1} is the true label (0 is legitimate, 1 is phishing) and p i is the predicted probability of phishing.

3. Results

3.1 Performance comparison

Table 2: Test set performance on the optimized 7-feature set for the models used. In the findings of the evaluation, Random Forest and XGBoost reached matching values when an F1-score of 0.952 and a ROC-AUC of 0.992 were achieved. Results show that MLP attained a level of success which was nearly the same since the F1-score reached 0.948. Even though the CNN-LSTM model carries a high level of complexity, the performance of the CNN-LSTM model stayed below the performance of the other models. These findings are consistent with the findings of recent studies into the performance of different models for detecting phishing sites, which have found that ensemble methods perform best for this task [23], [24], [25].

Table 3
Performance comparison of all models on the optimized 7-feature test set (Phishing=1)

Model	Accuracy	Precision	Recall	F1-Score	ROC-AUC
Logistic Regression	0.934	0.865	0.974	0.916	0.984
Random Forest	0.964	0.928	0.978	0.952	0.992
XGBoost	0.963	0.926	0.979	0.952	0.992
SVM (RBF)	0.943	0.880	0.980	0.927	0.980
MLP	0.960	0.918	0.980	0.948	0.991
CNN-LSTM Hybrid	0.949	0.912	0.956	0.933	0.988

Statistical significance: Paired McNemar tests were conducted by the research group on the test set of data to observe the outcome measurements. The results suggest that Random Forest and XGBoost show similar results because the analytical investigation revealed that Random Forest and XGBoost did not differ significantly in their performance ($p > 0.05$).

These specific models achieved better results than the MLP and CNN-LSTM models ($p < 0.01$ and $p < 0.001$ respectively) when the comparison was finalized. Figure 5 is a bar chart that compares the F1-scores and ROC-AUC of all six models. The F1-scores and ROC-AUC of Random Forest and XGBoost are tied at the highest level.

The 5-fold cross-validation results for all six models are presented in Table 4 for the reader to examine. Results show that the stability of gene-ralization is confirmed when Random Forest and XGBoost demonstrate very little variation in their output measurements. A high level of consistency was displayed by the computational frameworks during the testing phase while the assessment occurred. As noted in recent literature, deep learning approaches for phishing detection require careful regularization to avoid overfitting [26].

Figure 5
Bar chart comparing F1-score and ROC-AUC across all six models

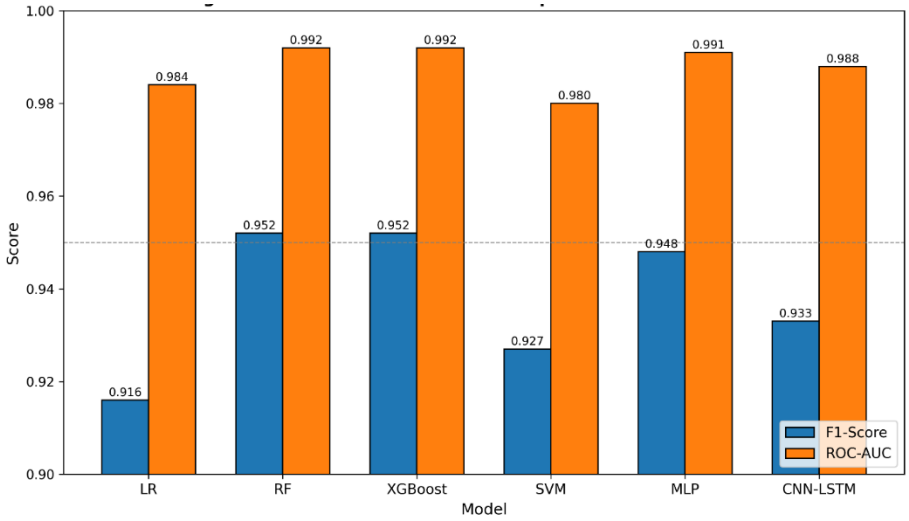


Figure 6
Comprehensive bar chart showing accuracy, precision, recall, F1-score, and ROC-AUC for all six models3.2. Cross-Validation and Stability

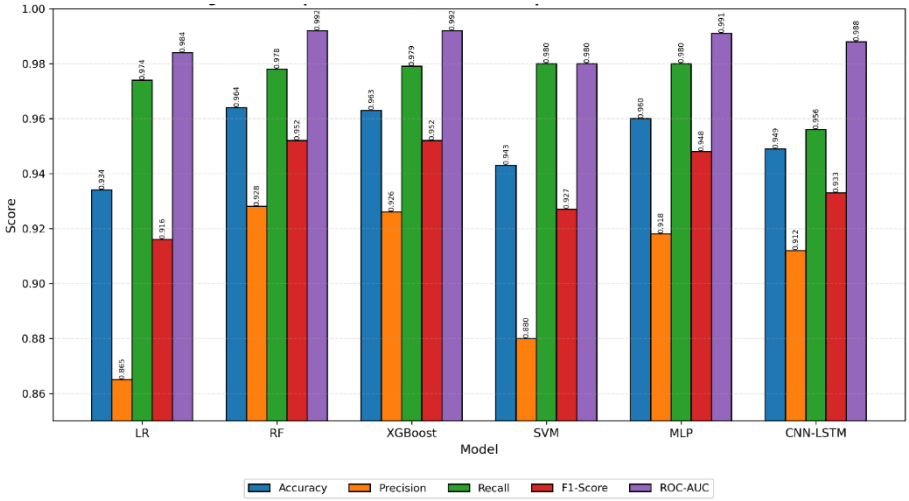


Table 4
5-fold cross-validation F1-score and accuracy for all six models (mean ± SD)

Model	CV F1-Score	CV Accuracy
Logistic Regression	0.9143 ± 0.0058	0.9325 ± 0.0049
Random Forest	0.9511 ± 0.0007	0.9628 ± 0.0006
XGBoost	0.9510 ± 0.0008	0.9627 ± 0.0007
SVM	0.9251 ± 0.0042	0.9412 ± 0.0038
MLP	0.9472 ± 0.0012	0.9591 ± 0.0010
CNN-LSTM	0.9315 ± 0.0035	0.9483 ± 0.0029

3.3. Confusion Matrix Analysis

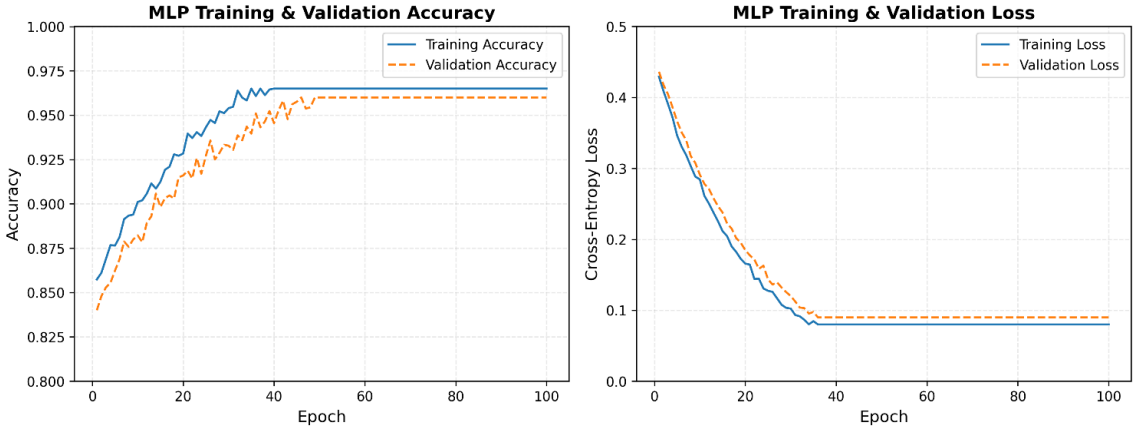
Figure 7 shows training/validation curves for the MLP model, confirming effective regularization. Confusion matrices (provided in supplementary materials) reveal:

- Random Forest: 796 false positives, 544 false negatives
- SVM: 1,568 false positives, 486 false negatives
- MLP: 910 false positives, 474 false negatives

Through the application of SHAP and LIME, better understanding of phishing detection models is

gained [27]. Analysis of precision-recall proves that Random Forest secures the most effective balance because Random Forest reduces wrong warnings while capturing fake websites at a high rate. Results show that these explainable AI techniques allow humans to see why the machine makes choices

Figure 7
Training and validation accuracy and loss curves for MLP over 100 epochs



The close alignment of training and validation curves confirms that dropout and L2 regularization successfully prevented overfitting.

3.4 External validation

To test the cross-dataset generalization, models learnt from the LegitPhish data set was tested on 11055 URL data samples from the UCI Phishing Websites Data Set. Only 7 out of the available lexical and structural features were used during the evaluation, since the two data sets do not contain a common set of features. These were the URL length, number of dots, number of hyphens, number of subdomains, URL entropy, top-level domain popularity or suspicious file extension indicators. The UCI dataset was not used for retraining. Rather, the models trained on the internal dataset were evaluated on the external dataset in a zero-shot scenario. This way, the robustness of the model to distribution shift is assessed more realistically.

As shown in Table 5, all models had a moderate drop in performance when compared to the LegitPhish test set. However, Random Forest and XGBoost achieved F1 scores in excess of 0.86 which suggests the decision boundaries learned had a reasonable degree of generalization outside the original distribution of the training set.

Table 5
Cross-Dataset Generalization Results

Model	LegitPhish Test F1	UCI Dataset F1 (Zero-shot)
Random Forest	0.952	0.873
XGBoost	0.952	0.869
MLP	0.948	0.851

The performance drop (≈8%) indicates room for improvement but confirms generalization beyond the training distribution. Real-time phishing detection systems based on ensemble learning have shown promise in cloud environments [28].

4. Discussion

This study yields several important findings.

First, pre-modeling data diagnostics are not optional they are imperative**.** Removing has ip address and https flag prevented illusory performance exceeding 0.98 accuracy. A model's validity depends not on high metrics alone but on the integrity of the signals it learns. This problem is pervasive in cybersecurity ML, where datasets often contain unique, non-global biases [23], [24]. Adversarial attacks on phishing detection systems

have also been shown to exploit such dataset-specific artifacts [29].

Second, Research indicates that deep learning is equaled or beaten by ensemble tree-based methods when working with structured tabular data. Identical highest success measurement (F1=0.952) was reached by Random Forest and XGBoost, even though MLP followed very closely (F1=0.948). It has been observed that ensemble tree-based methods maintain high accuracy for this data type. The CNN-LSTM hybrid added no value despite higher complexity, as it searched for non-existent spatial or sequential patterns in independent features a fundamentally misaligned inductive bias [26]. This challenges the overemphasis on complex architectures for marginal gains. Recent large-scale datasets for URL-based phishing detection have enabled more robust benchmarking of these approaches [30], [31].

Third, For the purpose of real-world implementation, the most useful balance is provided by Random Forest. The precision score of Random Forest (0.928) stays above MLP (0.918) and stays much higher than SVM (0.880). Because false positives like stopping real web addresses damage the way individuals who use the software trust the technology, these errors are more harmful than missing a fake site. The results suggest that the 796 false positives from Random Forest show a clear operational benefit when compared to the 1,568 errors from SVM.

Table 6
Inference Time Comparison

Model	Inference Time (ms)
Logistic Regression	0.08
Random Forest	0.21
XGBoost	0.24
SVM	0.35
MLP	0.52
CNN-LSTM	1.87

Results show that Random Forest remains appropriate for immediate categorization activities because processing takes less than 1 ms for every URL.

At the default classification threshold of 0.5, the Random Forest model achieved an F1-score of 0.952, precision of 0.928, and recall of 0.978 (Table 2). Adjusting the threshold changes the trade-off between precision and recall. Increasing the threshold to 0.6 increases precision to 0.951 but reduces recall to 0.941 (a drop of 0.037). Decreasing the threshold to 0.4 increases recall to 0.989 but reduces precision to 0.887. The optimal threshold depends on the operational cost of false positives versus false negatives in a deployed system.

Limitations include reliance on a single primary dataset (though external validation was added), static URL analysis only, and lack of adversarial robustness testing. Future work should integrate multi-modal signals (landing page content, WHOIS data, traffic patterns) and test against adversarial examples [28], [29].

5. Conclusion

This study was a detailed assessment of various machine learning techniques for detecting phishing URLs with the LegitPhish dataset. Emphasis was placed on pre-modeling data diagnostics in order to remove highly correlated variables and shortcut features present in the data that could enhance the model performance artificially.

The results show that ensemble techniques based on trees gave the best performance overall on the optimized seven-feature dataset. Random Forest and XGBoost models performed equally, with an F1-scores of 0.952 and ROC-AUC of 0.992, better than classical machine learning models and even higher performing deep learning architectures. While the MLP model was competitive, the CNN-LSTM architecture did not offer any advantages for structured tabular URL features.

The findings demonstrate the need to be rigorous in feature analysis and leakage prevention in the cybersecurity machine-learning research community. They also propose that their well-balanced ensemble methods can generate high accuracy, low complexity, and real-time Solutions for phishing URL detection.

Acknowledgments

This work was supported by Al-Farabi Kazakh National University and the International Information Technology University, Almaty, Kazakhstan.

Funding

This research received no external funding.

Conflicts of Interest

The authors declare no conflict of interest.

Author Contributions

Conceptualization, A.A. and H.M.; Methodology, A.A.; Software, H.M.; Validation, A.A. and H.M.; Formal Analysis, A.A.; Investigation, A.A.; Resources, H.M.; Data Curation, A.A.; Writing – Original Draft, A.A.; Writing – Review & Editing, H.M. and A.A.; Visualization, A.A.; Supervision, H.M.; Project Administration, A.A.

References

1. R. Heartfield and G. Loukas, "A taxonomy of attacks and a survey of defence mechanisms for semantic social engineering attacks," *ACM Comput. Surv.*, vol. 48, no. 3, pp. 1–39, 2016, doi: 10.1145/2835372.
2. Anti-Phishing Working Group (APWG), "Phishing activity trends report, 4th quarter 2022," Mar. 2023. [Online]. Available: https://apwg.org/trendsreports/
3. A. Oest, P. Zhang, B. Wardman, *et al.*, "Sunrise to sunset: Analyzing the end-to-end life cycle of phishing attacks," in *Proc. USENIX Security Symp.*, 2020, pp. 361–378.
4. A. Basit, M. Zafar, A. R. Javed, and M. Rizwan, "A comprehensive survey of AI-enabled phishing detection techniques," *J. Cybersecur. Priv.*, vol. 1, no. 4, pp. 42–60, 2020, doi: 10.3390/jcp1040004.
5. O. K. Sahingoz, E. Buber, O. Demir, and B. Diri, "Machine learning based phishing detection from URLs," *Expert Syst. Appl.*, vol. 117, pp. 345–357, 2019, doi: 10.1016/j.eswa.2018.09.029.
6. R. M. Mohammad, F. Thabtah, and L. McCluskey, "Predicting phishing websites based on self-structuring neural network," *Neural Comput. Appl.*, vol. 26, pp. 443–455, 2015, doi: 10.1007/s00521-014-1690-1.
7. S. Smadi, N. Aslam, L. Zhang, and M. Alhabeeb, "A comparative study of machine learning classifiers for phishing detection," *J. Inf. Secur. Appl.*, vol. 54, p. 102593, 2020, doi: 10.1016/j.jisa.2020.102593.
8. R. Zieni, L. Massari, and M. C. Calzarossa, "Phishing or not phishing? A survey on the detection of phishing websites," *IEEE Access*, vol. 11, pp. 18499–18519, 2023, doi: 10.1109/ACCESS.2023.3247135.
9. S. Asiri, Y. Xiao, S. Alzahrani, S. Li, and T. Li, "A survey of intelligent detection designs of HTML URL phishing attacks," *IEEE Access*, vol. 11, pp. 6421–6443, 2023, doi: 10.1109/ACCESS.2023.3237798.
10. T. Peng, I. Harris, and Y. Sawa, "Detecting phishing attacks using natural language processing and recurrent neural networks," *J. Netw. Comput. Appl.*, vol. 176, p. 102942, 2021, doi: 10.1016/j.jnca.2020.102942.
11. P. Yang, G. Zhao, and Y. Liu, "Phishing detection using attention-based bidirectional LSTM and convolutional neural network," *IEEE Internet Things J.*, vol. 8, no. 5, pp. 3352–3363, 2021, doi: 10.1109/JIOT.2020.3022025.
12. Y. Cao, S. Li, and Y. Liu, "PhishTransformer: A transformer-based model for detecting phishing URLs," *Comput. Secur.*, vol. 127, p. 103102, 2023, doi: 10.1016/j.cose.2023.103102.
13. S. Gupta and A. Sharma, "Robust phishing detection in the presence of adversarial examples," *Int. J. Crit. Infrastruct. Prot.*, vol. 38, p. 100536, 2022, doi: 10.1016/j.ijcip.2022.100536.
14. R. Potpelwar, "LegitPhish dataset," Mendeley Data, V2, 2025, doi: 10.17632/hx4m73v2sf.2.
15. R. Potpelwar, U. V. Kulkarni, and J. M. Waghmare, "LegitPhish: A large-scale annotated dataset for URL-based phishing detection," *Data in Brief*, vol. 63, p. 111972, 2025, doi: 10.1016/j.dib.2025.111972.
16. M. Medelbekov, M. Nurtas, and A. Altaibek, "Machine learning methods for phishing attacks," *J. Probl. Comput. Sci. Inf. Technol.*, vol. 1, no. 2, 2023, doi: 10.26577/JPCSIT.2023.v1.i2.02.
17. A. Jabr Saleh Albahadili, A. Akbas, and J. Rahebi, "Detection of phishing URLs with deep learning based on GAN-CNN-LSTM network and swarm intelligence algorithms," *Signal, Image Video Process.*, vol. 18, pp. 4979–4995, 2024, doi: 10.1007/s11760-024-03180-5.
18. D. Sugianto, R. A. Putawa, C. Izumi, and S. A. Ghaffar, "Uncovering the efficiency of phishing detection: An in-depth comparative examination of classification algorithms," *Int. J. Appl. Inf. Manag.*, vol. 4, no. 1, pp. 22–29, Apr. 2024, doi: 10.47738/ijaim.v4i1.72.
19. R. Basnet, S. Mukkamala, and A. H. Sung, "Detection of phishing attacks: A machine learning approach," in *Studies in Fuzziness and Soft Computing*, vol. 226, 2008, pp. 373–383, doi: 10.1007/978-3-540-77465-5 19.
20. S. Mittal and P. K. Das, "A novel ensemble method for phishing detection using stacking of machine learning classifiers," *Expert Syst. Appl.*, vol. 184, p. 115545, 2021, doi: 10.1016/j.eswa.2021.115545.
21. F. Alotaibi and A. Alshamrani, "A hybrid deep learning model for phishing detection using convolutional and recurrent neural networks," *J. King Saud Univ. – Comput. Inf. Sci.*, vol. 34, no. 8, pp. 5210–5220, 2022, doi: 10.1016/j.jksuci.2021.06.006.
22. A. Zamir, H. U. Khan, T. Iqbal, and M. M. Yousaf, "Phishing detection: A comparative analysis of machine learning algorithms," *Int. J. Adv. Comput. Sci. Appl.*, vol. 12, no. 4, pp. 1–9, 2021, doi: 10.14569/IJACSA.2021.0120265.
23. S. Salloum, T. Gaber, S. Vadera, and K. Muhammad, "A systematic review of machine learning techniques for phishing detection," *IEEE Access*, vol. 10, pp. 65729–65758, 2022, doi: 10.1109/ACCESS.2022.3183925.
24. V. Ravi and R. Chaganti, "A comprehensive review on phishing detection using deep learning techniques," *Arch. Comput. Methods Eng.*, vol. 29, pp. 501–525, 2022, doi: 10.1007/s11831-021-09589-4.
25. A. A. Alqarni and M. A. Alqarni, "A systematic literature review on phishing website detection techniques," *J. King Saud Univ. – Comput. Inf. Sci.*, vol. 35, no. 2, pp. 590–611, 2023, doi: 10.1016/j.jksuci.2023.01.004.

26. N. Q. Do, A. Selamat, O. Krejcar, E. Herrera-Viedma, and H. Fujita, "Deep learning for phishing detection: Taxonomy, challenges and future directions," *IEEE Access*, vol. 10, pp. 36429–36464, 2022, doi: 10.1109/ACCESS.2022.3151903.

27. A. Jain and V. Singh, "Explainable AI for phishing detection: A case study with SHAP and LIME," *Comput. Secur.*, vol. 124, p. 102956, 2023, doi: 10.1016/j.cose.2022.102956.

28. M. Williams and M. Tully, "Real-time phishing detection using ensemble learning in cloud environments," *J. Cloud Comput.*, vol. 10, pp. 1–18, 2021, doi: 10.1186/s13677-021-00252-8.

29. J. Hong, T. Kim, J. Liu, and N. Park, "Adversarial machine learning in phishing detection: Attacks and defenses," *IEEE Trans. Dependable Secure Comput.*, vol. 19, no. 5, pp. 3124–3138, 2021, doi: 10.1109/TDSC.2021.3064210.

30. D. M. Linh and T. C. Hung, "A feature-engineered dataset of benign and phishing URLs for machine learning and large language models evaluation," *Data in Brief*, vol. 63, p. 112162, 2025, doi: 10.1016/j.dib.2025.112162.

31. A. N. Alharbi and W. G. Alghamdi, "Phishing website detection using machine learning: A comprehensive survey," *Electronics*, vol. 11, no. 22, p. 3761, 2022, doi: 10.3390/electronics11223761.

Information about the authors:

Aerna Aheti is a master's student at the Department of Artificial Intelligence and Big Data, Al-Farabi Kazakh National University. Her research focuses on ensemble machine learning methods for cybersecurity applications, with particular emphasis on phishing URL detection and comparative model evaluation. She has extensive experience in data diagnostics, feature selection, and leakage prevention in cybersecurity ML pipelines. Her current work involves optimizing Random Forest and XGBoost classifiers for real-time threat detection systems (Almaty, Kazakhstan).

Hadi Mukhtar is a master's student at the Department of Artificial Intelligence and Big Data, Al-Farabi Kazakh National University. His research focuses on deep learning architectures for security-related classification tasks, including CNN-LSTM hybrid models for phishing detection and network anomaly identification. He specializes in hyperparameter optimization, model regularization techniques, and cross-dataset generalization assessment for deep neural networks in cybersecurity contexts (Almaty, Kazakhstan, ORCID iD: 0009-0009-8252-779X).

Submission received: 26 January, 2026
Revised: 18 March, 2026
Accepted: 18 March, 2026

IRSTI 28.23.01 https://doi.org/10.26577/jpcsit4220268

Aidana Karibayeva , Balzhan Abduali*

Al-Farabi Kazakh National University, Almaty, Kazakhstan
*e-mail: abduali.balzhan@kaznu.kz

AI-DRIVEN TEXT AND AUDIO SYNTHESIS FOR LOW-RESOURCE LANGUAGE DATASETS

Abstract. This paper presents an AI-driven methodology for constructing parallel text corpora and synthesizing parallel audio datasets for low-resource Turkic language pairs, namely Kazakh-Kyrgyz and Kazakh-Uzbek. The scarcity of high-quality linguistic and speech resources for these languages poses significant challenges for the development of neural machine translation and automatic speech recognition systems. The paper proposes and validates a methodology for parallel dataset construction driven by artificial intelligence, in which the selection of a translation system is guided by a set of criteria encompassing free accessibility, translation quality, and processing efficiency, assessed through experiments on a 2000-sentence test set. Based on this evaluation, large-scale parallel corpora for the Kazakh-Kyrgyz and Kazakh-Uzbek language pairs were generated using the selected AI system. A subsequent manual error analysis revealed that approximately 3.7% (Kazakh-Kyrgyz) and 4.3% (Kazakh-Uzbek) of the translations contained inaccuracies, indicating the need for post-editing and further model refinement. Audio synthesis experiments using MMS-TTS and TurkicTTS systems demonstrated that synthetic speech of near-natural quality can be generated for both languages, with NISQA scores reaching up to 4.44 for Uzbek. The findings confirm that the proposed methodology provides a practical and scalable foundation for expanding linguistic resources for low-resource Turkic languages and supporting further research in machine translation and speech processing.

Keywords: low-resource languages, methodology for creating datasets, parallel text corpora, AI systems, audio synthesis.

1. Introduction

There are more than 7,000 languages spoken worldwide, yet the majority of digital language technologies are developed for only a small subset of them [1]. While more than half of the world's population uses fewer than 20 major languages, a large number of languages remain underrepresented in the digital space. Many of these so-called low-resource languages are official languages of countries with relatively small populations, including several states in Central Asia such as Kazakhstan. The scarcity of digital and linguistic resources severely hampers the advancement of modern language technologies for these languages.

Low-resource languages often suffer from digital exclusion due to the lack of essential linguistic resources required for effective automatic language processing. The development of natural language processing (NLP) systems typically requires the availability of character encoding standards, lexical resources, and well-described linguistic rules, including grammar, syntax, and morphology. Although the problems associated with character encoding and the coverage of a core vocabulary have been partially solved by modern information technologies, the creation of comprehensive linguistic resources remains a difficult and labor-intensive task. Consequently, these languages suffer from an acute deficiency of extensive annotated datasets, lexicons, NLP toolkits, and digitized textual materials required for model development and evaluation. Compared to well-resourced languages like English, Chinese, Hindi, Spanish, or French, such insufficient data makes it considerably more difficult to construct reliable systems capable of handling machine translation, speech recognition, text classification tasks [2–4].

Kyrgyz and Uzbek fall into the category of low-resource languages, primarily due to the insufficient availability of high-quality linguistic data required to train contemporary natural language processing and machine translation systems. Among the most pressing challenges is the restricted access to parallel corpora for these languages. Although research interest in building and benchmarking machine translation systems for under-resourced Turkic languages has grown considerably in recent years [5, 6], the absence of substantial high-quality parallel datasets continues to be a fun-

damental bottleneck impeding further advances in this domain.

Several approaches have been proposed to address this problem. One solution involves the creation of synthetic corpora. For example, by constructing sentences based on a complete set of morphological rules and suffixes [7, 8]. One of the most widely used approaches today is to create synthetic parallel data by translating monolingual texts from one language to another using machine translation. However, this approach can also lead to significant difficulties in terms of translation quality. Machine translation systems for resource-constrained languages lack accuracy due to the limited amount of training data. Consequently, this may give rise to semantic distortions, grammatical inaccuracies, and incorrect interpretations of phrases, ultimately compromising the overall quality of the resulting parallel corpus.

A common approach to building synthetic parallel corpora for low-resource languages involves using an intermediate language, typically English [9, 10]. While this method helps compensate for the lack of direct translation data, it is not always effective for closely related language pairs such as Turkic languages. Routing translation through English often results in the loss of semantic and grammatical nuances specific to these languages, thereby reducing the quality of the generated data [11]. This highlights the need for direct corpus construction methods that better preserve the linguistic properties of such language pairs.

Publicly available parallel data for the Kyrgyz-Kazakh and Uzbek-Kazakh language pairs remains extremely scarce, making it difficult to develop and train effective neural machine translation (NMT) models. Research and openly accessible resources for these language pairs are also notably limited [12], underscoring the need for systematic approaches to constructing high-quality parallel corpora.

This article presents an AI-driven methodology for constructing parallel text corpora for the Kyrgyz-Kazakh and Uzbek-Kazakh language pairs, directly addressing the data scarcity challenges outlined above. The proposed approach leverages artificial intelligence techniques for both text and audio synthesis, aiming to enrich language resources for low-resource Turkic languages and to support further research in machine translation and related natural language processing tasks.

2. Materials and methods

This section outlines the methodological foundations of the proposed AI-driven approach to constructing parallel datasets for low-resource Turkic languages. It begins by reviewing the core challenges associated with data scarcity and examines existing resources and strategies for parallel corpus construction. Next, it discusses the role of modern AI-based systems in the automatic generation of both text and audio data, and motivates the selection of an AI-assisted pipeline for dataset creation. Finally, the section introduces the overall methodology adopted in this study for building and evaluating parallel corpora for the Kyrgyz-Kazakh and Uzbek-Kazakh language pairs.

2.1 Data Scarcity in Low-Resource Languages and Approaches to Parallel Corpus Creation

The scarcity of large, high-quality parallel text datasets represents one of the most significant obstacles for low-resource languages. Turkic languages including Kazakh, Kyrgyz, and Uzbek are particularly affected, given that openly accessible bilingual resources for these languages remain considerably limited [13, 14]. Efforts to bridge this gap have included developing bilingual dictionaries and compiling small parallel datasets for pairs such as Uyghur–Kazakh, Kazakh–Kyrgyz, and Kyrgyz–Uyghur, alongside pivot-based translation strategies that rely on English as a bridge language [14, 15]. Despite these efforts, the available resources fall short in both volume and linguistic coverage, rendering them insufficient for training robust neural machine translation systems.

Creating parallel data for languages with limited resources is a complex and multifaceted process that may involve various sources and methods. Existing parallel datasets represent the most straightforward option, and a number of large multilingual resources are available, including OPUS, TED Talks subtitles, and Europarl. OPUS stands out as one of the most extensively utilized open-access repositories of multilingual parallel corpora among the available resources [16–18]. Such datasets serve as a fundamental resource for training and evaluating machine translation systems, as well as for advancing research in multilingual natural language processing. Nevertheless, for a number of Turkic language pairs – notably Kazakh–Kyrgyz and Kazakh–Uzbek – the available resources remain inadequate in both

coverage and scale to effectively support the practical development of machine translation systems [19, 20].

Recent advances in neural machine translation and large multilingual models, such as attention-based NMT architectures, XLM-R, and the NLLB project, have substantially enhanced translation quality for low-resource languages through the utilization of large-scale multilingual pre-training and cross-lingual transfer learning techniques [21–24]. At the same time, these developments have made it possible to use machine translation systems not only as end-user tools but also as instruments for data generation, enabling the construction of synthetic parallel corpora from monolingual texts.

Several studies have discussed different strategies for corpus construction, including the use of open data sources, crowdsourcing, proprietary datasets, and human–machine collaboration [25-27]. In particular, AI-assisted and semi-automatic approaches have gained increasing attention due to their scalability and cost-effectiveness. However, the quality of automatically generated parallel data remains a critical issue, especially for low-resource languages, and typically requires careful selection of translation systems as well as subsequent quality assessment and filtering.

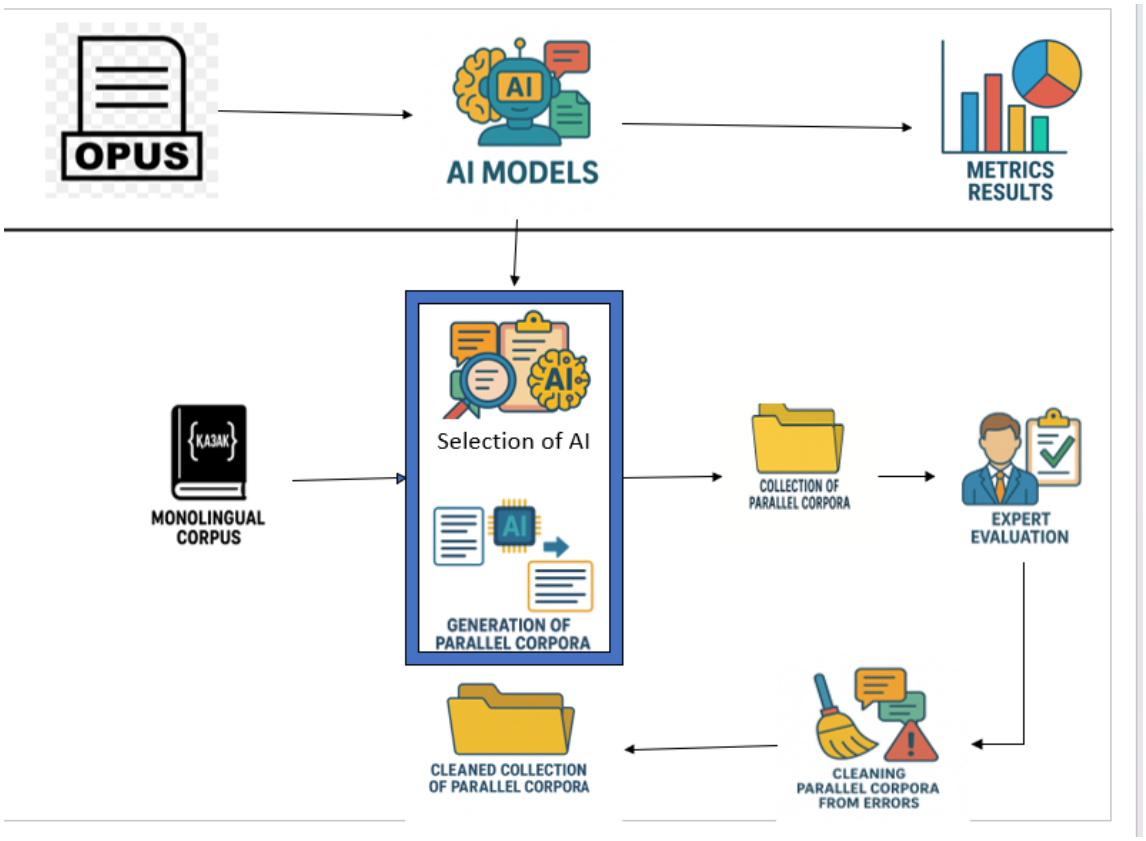
Building on these considerations, this study proposes a methodology that leverages artificial intelligence to construct parallel text corpora for the Kazakh–Kyrgyz and Kazakh–Uzbek language pairs. The approach seeks to integrate the scalability of contemporary machine translation systems with rigorous quality evaluation, with the ultimate goal of producing practical datasets suitable for training and benchmarking neural machine translation models.

2.2 Developed methodology

2.2.1 AI-Assisted Parallel Text Corpus Construction Methodology

As depicted in Figure 1, the methodology put forward in this study comprises three core phases: (1) preparation, (2) selection of an appropriate AI system, and (3) corpus construction.

Figure 1
Architecture of the proposed method for text



The preparation stage consists of several initial tasks. First, an extensive review of previous studies is carried out to investigate recent developments in machine translation for low-resource languages, with special attention given to Turkic language pairs such as Kazakh–Kyrgyz and Kazakh–Uzbek. Next, publicly available parallel corpora for these language pairs are collected from open resources, particularly the OPUS platform, followed by an analysis of their reliability and overall quality. At the same time, existing neural machine translation approaches and AI-driven translation systems applicable to the selected languages are examined. Based on the outcomes of the literature analysis, suitable translation evaluation measures are determined. Furthermore, a number of criteria for selecting AI systems are established, including translation performance, model design, computational requirements, and the accessibility of training resources. Finally, a balanced and domain-diverse benchmark dataset of parallel sentences is prepared to support an objective assessment of translation quality.

Several AI-based translation systems were evaluated against a set of predefined criteria, including accessibility, translation quality, and processing efficiency. The systems considered for comparison were Google Translate, NLLB, ChatGPT, DeepSeek, GitHub Copilot, and Gemma [32–37], which can be broadly divided into specialized translation systems and general-purpose large language models. Each system was used to translate a test set of Kazakh sentences into Kyrgyz and Uzbek, and the resulting outputs were assessed using the metrics described in Section 2.4. The system demonstrating the best overall performance in terms of translation quality, efficiency, and accessibility was subsequently selected for large-scale corpus generation.

The second stage is devoted to the comparative selection of AI systems. During this process, every candidate model translates the Kazakh portion of the evaluation dataset into Kyrgyz and Uzbek. The generated outputs are then assessed using the previously chosen evaluation metrics. Considering both the metric results and the predefined selection conditions, the candidate systems are systematically compared and filtered through several evaluation rounds until the most effective model is selected for further large-scale synthetic data generation.

The final stage focuses on corpus construction. At this stage, the chosen AI system is applied to translate the Kazakh component of an existing Kazakh–English parallel corpus into Kyrgyz and Uzbek. As a result, extensive synthetic parallel datasets for the Kazakh–Kyrgyz and Kazakh–Uzbek language pairs are created. These newly generated corpora can later serve as training and evaluation resources for neural machine translation systems.

2.2.2 Text-based audio synthesis

This study uses a cascade methodology for creating a text-based audio corpus. It aims to create a parallel text-audio corpus for Turkic languages. The source language is Kazakh, and the target languages are Uzbek and Kyrgyz.

The proposed approach differs from the traditional speech data collection and automatic recognition methodology. In the classical method, the quality of the corpus depends on the speakers, the writing environment, dialect features, and the accuracy of ASR/STT systems. In low-resource languages, this leads to transcription errors, phonetic distortions, and lower data quality.

Based on high-quality audio and text in the input language (Kazakh), first, supervised parallel text data is generated, and then converted into synthetic speech using TTS systems. This approach increases the scalability, repeatability, and controllability of the corpus creation process.

The stages of the proposed methodology consist of the following:

1 Kazakh data preparation. At this stage, the source texts in Kazakh are preprocessed: normalized, standardized, and unnecessary elements removed. Morphological and syntactic features are taken into account. This text is taken from the original high-quality audio dataset from NU corpora.

2 AI- based machine translation. The transcriptions are translated into Uzbek and Kyrgyz using the NLLB neural translation model. If necessary, additional grammatical and syntactic corrections are made, as in the previous text data processing section.

3. Speech synthesis – Kazakh, Uzbek, and Kyrgyz texts are converted into audio files using TTS systems. For each sentence, a corresponding "text – audio" pair is created.

In the study, only a limited number of TTS systems were identified that support the Kazakh language and our target languages (Uzbek and Kyrgyz). These systems are: MMS-TTS and TurkicTTS. The TTS systems were evaluated according to the following criteria: support for the target languages (Uzbek and Kyrgyz), availability under an open license, local deployment, scalability, naturalness, and suitability for generating a synthetic speech corpus. During the work, for 2 languages, the Kyrgyz

Based on the analysis, MMS-TTS and TurkicTTS were selected for the study. Although there are also systems for the Turkic languages, CoquiTTS and ElevenLabs, they were excluded because they did not fully meet the methodological requirements.

The CoquiTTS model does not support the Kyrgyz language. Therefore, additional training on a fixed Kyrgyz speech corpus would be required to use CoquiTTS in this study. Since such a corpus was not used in the study, the system was excluded from the analysis. As an additional limitation, the discontinuation of the Coqui company reduces the reliability of the system's long-term support. ElevenLabs, on the other hand, supports many languages, including Kyrgyz and Uzbek, and provides high-quality speech synthesis. However, the platform is commercial and closed. It does not offer open-source code, does not support local deployment, and limits the amount of generation available through tariff plans. In addition, dependence on external APIs reduces the reproducibility of experiments. For these reasons, ElevenLabs was excluded as a system that does not meet the requirements of openness, scalability, and independent scientific use.

MMS-TTS was selected as an open, multilingual system supporting more than 1,100 languages, including Uzbek and Kyrgyz. Its local deployment, lack of commercial restrictions, and ability to automatically generate audio data make it suitable for working with low-resource Turkic languages. In addition, the use of phonemic representations across languages increases their applicability to languages with limited speech resources.

TurkicTTS was selected as a special speech synthesis system for Turkic languages. Its advantage is its focus on the phonetic and morphological features of this language group, including agglutinativity, pronunciation, and word-form variability. Support for Uzbek and Kyrgyz languages, as well as the ability to further train on user data, make TurkicTTS suitable for the tasks of this study. Thus, MMS-TTS and TurkicTTS were selected as the most suitable systems for synthetic speech generation in Uzbek and Kyrgyz languages. One is a universal multilingual model, the other is a system adapted to Turkic languages. Using both together increases the study's objectivity.

4. Corpus formation – All texts and audio files are combined into a single structure:

{text KAZ} → {text UZB/KYR} → {audio KAZ (high-quality human voice)} → {audio UZB/KYR (synthetic)}

The advantages of the proposed methodology are as follows:

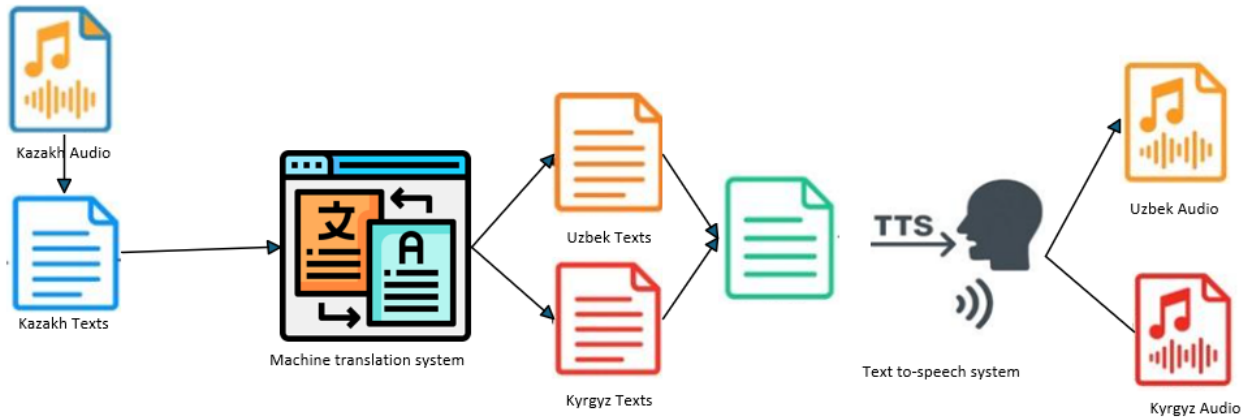
- High data quality control.
- Independent of narrators and recording conditions.
- Easy to scale and repeat.
- Reduces data collection costs and reduces transcription errors.

The proposed *text-based audio synthesis* methodology allows for the efficient and reliable formation of a parallel text-audio corpus in the Turkic language

An illustrative diagram of the proposed methodology is presented in Figure 2.

Figure 2

The scheme of Text-based audio synthesis Generation for Uzbek and Kyrgyz languages



2.3 Parallel Data Sources and Preprocessing

To conduct the experimental study, parallel text datasets were obtained from publicly available corpus resources. A review of existing multilingual corpora indicated that datasets suitable for Kazakh–Kyrgyz and Kazakh–Uzbek machine translation remain limited in both quantity and quality. The main available corpora from the OPUS collection are summarized in Table 1.

Table 1
Available OPUS Parallel Corpora for Kazakh–Kyrgyz and Kazakh–Uzbek

Name of Dataset	Parallel sentences	Dataset link
OPUS [15]	214 183 KZ-KG	https://opus.nlpl.eu/corpora-search/kk&ky
OPUS [15]	249 941 KZ-UZ	https://opus.nlpl.eu/corpora-search/kk&uz

According to these resources, 2000 Kazakh-Kyrgyz and Kazakh-Uzbek sentence pairs were taken to create a test set. Kyrgyz and Uzbek sentences were considered as gold standard references, and Kazakh sentences were used as source texts for translation and evaluation.

In addition to building large parallel corpora for low-resource Turkic languages, a monolingual Kazakh dataset was selected as the source material. The dataset was obtained from an open-access resource and originally contained ∽300 thousands parallel Kazakh–English sentence pairs [28]. In the context of this study, only the Kazakh side of the corpus was selected and utilized as the source input for automatic translation. Using the artificial intelligence system selected here, this corpus was translated into Kyrgyz and Uzbek, resulting in newly created Kazakh-Kyrgyz and Kazakh-Uzbek parallel corpora.

This approach allows for the efficient creation of large parallel datasets for low-resource language pairs by using monolingual or bilingual resources and modern artificial intelligence-based translation systems.

2.4 Quality metrics

2.4.1 Translation Quality Metrics for Text

Machine translation quality was evaluated using several complementary metrics, since a single evaluation method cannot fully reflect translation performance, especially for morphologically rich Turkic languages. The selected metrics measure translation quality from different perspectives, including lexical similarity, character-level correspondence, editing distance, and semantic adequacy.

Among the applied metrics, BLEU evaluates the overlap between generated and reference translations using n-gram matching and remains one of the most widely used metrics in machine translation research [29]. In contrast, WER focuses on the number of editing operations required to transform the generated sentence into the reference text, where lower values indicate better performance. To better capture morphological variations common in Kazakh, Kyrgyz, and Uzbek, the ChrF metric was additionally employed. Unlike word-based metrics, ChrF operates at the character level, making it more sensitive to agglutinative language structures [30]. Semantic and linguistic aspects of translation quality were assessed using METEOR and COMET. METEOR considers factors such as stemming, synonym matching, and word order, while COMET applies neural-based evaluation techniques to estimate translation fluency and semantic consistency [31].

2.4.2 Metrics for synthesized speech

To obtain a more comprehensive evaluation of synthesized speech quality, several objective assessment metrics were employed.

MOSNet was used for automatic prediction of perceptual speech quality scores comparable to the Mean Opinion Score (MOS).

DNSMOS evaluates the overall audio quality, naturalness of speech, and the level of distortion in the generated speech signal. Additionally, a neural network-based quality assessment was conducted using NISQA. Furthermore, SSL-MOS, based on self-learning representations, was used to assess the naturalness and intelligibility of synthesized speech.

3. Results

3.1 Text Corpus Generation Results

Table 2 presents the evaluation results of six AI-based translation systems on a test set of 2,000 Kazakh sentences from the OPUS datasets, comparing their performance across multiple metrics for both Kazakh–Kyrgyz and Kazakh–Uzbek language pairs.

Table 2
Translation model performance assessed on a test set of 2,000 Kazakh sentences drawn from the OPUS datasets

Metrics / System	BLEU		WER		METEOR		COMET		ChrF	
	kg	uz	kg	uz	kg	uz	kg	uz	kg	uz
Google Translator	12.00	13.00	0.93	0.91	0.08	0.06	0.70	0.71	23.02	24.41
Chat GPT	**37.60**	**37.30**	0.87	0.86	**0.16**	**0.16**	**0.91**	**0.89**	31.57	**32.78**
Nllb-200-3.3	30.40	31.40	0.88	**0.85**	0.13	0.13	0.78	0.72	27.12	25.35
DeepSeek	34.00	33.30	**0.88**	0.87	0.15	0.15	0.83	0.80	**31.58**	32.51
Copilot	27.20	25.00	0.89	0.88	0.15	0.13	0.82	0.76	31.40	30.70
Gemma-2-27b	32.00	32.00	0.87	0.87	0.14	0.14	0.81	0.81	30.27	31.36

The Figure 1 visually illustrates the performance differences between the evaluated translation systems, enabling a clear comparison of their effectiveness and facilitating the selection of the most suitable model.

Table 3 illustrates the translation time required by each AI system to process 300 thousand Kazakh sentences, revealing a substantial difference in efficiency between Gemma and NLLB, which completed the same task in approximately 70 days and 3 days, respectively.

Figure 1
Comparison of translation systems

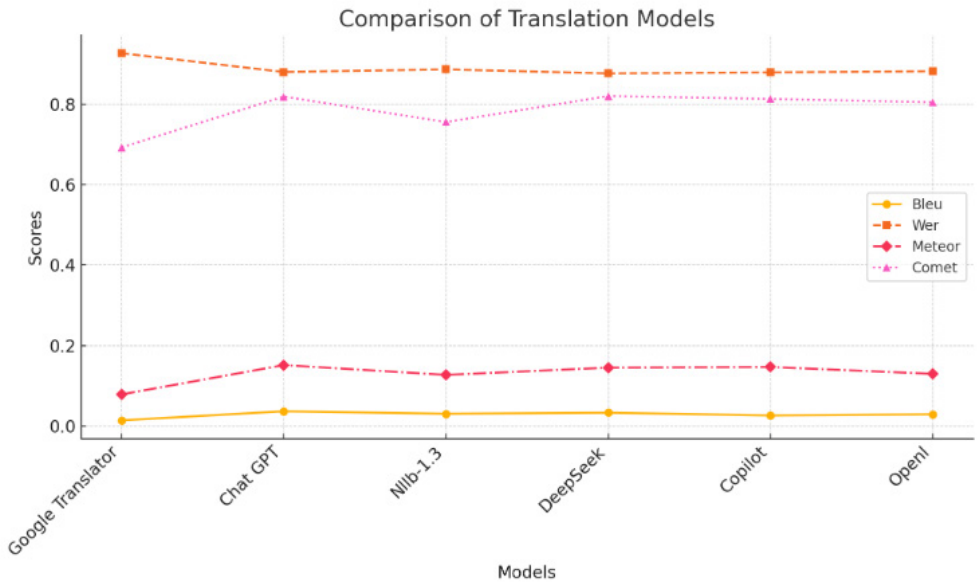


Table 3
Processing Time Required by Each AI System for Large-Scale Text Translation

Models	Estimated processing time required to translate a corpus of 300 thousand sentences
Gemma	~70 days
NLLB	~3 days

Table 4
Summary of Translation Error Analysis for Kazakh–Kyrgyz and Kazakh–Uzbek Language Pairs

Indicator	Kazakh–Kyrgyz	Kazakh–Uzbek
Total errors identified	11175	12930
Error rate	3.7%	4.3%
Main error types	Abbreviation, Repetition	Abbreviation, Repetition

The table 4 presents information about errors and quantity of errors. Manual inspection of the translated corpus revealed two predominant error categories: semantic errors, characterized by incorrect conveyance of meaning, and lexical errors, involving missing or inaccurately translated words. The total number of identified errors amounted to 11175 for the Kyrgyz and 12930 for the Uzbek translations. Addressing these errors led to a notable enhancement in the overall quality of the parallel corpus. Based on these figures, the estimated error rate stands at approximately 3.7% for the Kazakh–Kyrgyz and 4.3% for the Kazakh–Uzbek language pairs, encompassing inaccuracies in abbreviation translation and repetitions that were subsequently removed using a specially developed automated program.

3.2 Speech synthesis results

Based on the *proposed methodology for text-based audio synthesis*, a parallel corpus was obtained, containing the original Kazakh-language data and their Uzbek and Kyrgyz versions. The compiled data were uploaded to the Hugging Face platform. The test set of 1000 audio files is available at the following link: https://huggingface.co/datasets/KazNU-Lab/audio-parallel-1k-mms.

For the experimental section, 500 audio fragments of each male and female voice were selected from the NU Corpus. The dataset for the overall testing consisted of 1,000 audio recordings. This data was used for reference-free metrics that do not require a ground-truth recording to evaluate synthetic speech quality. They allow for the analysis of speech naturalness, intelligibility, and overall perceived quality.

Table 5
Results of synthetic audio generation for Uzbek

Language	DNSMOS	MOSNet	NISQA	SSL-MOS
MMS	3.33	4.04	4.26	3.24
TurkicTTS	3.31	3.99	4.44	2.39

Table 6
Results of synthetic audio generation for Kyrgyz

Language	DNSMOS	MOSNet	NISQA	SSL-MOS
MMS	3.10	3.75	4.00	2.60
TurkicTTS	3.15	3.80	4.20	1.90

The MMS system for Uzbek showed high results on the SSL-MOS metric (2.39 for TurkicTTS). This difference indicates that the MMS system's speech is prosodically (intonation, stress, speech tempo, pauses between words or phrases, rhythm, pitch / F0, speech intensity, emotional coloration) stylistically close to natural speech. Such a property is crucial for training ASR systems.

The TurkicTTS system for Uzbek has the advantage according to the NISQA metric (TurkicTTS 4.44, MMS 4.26). This indicates a high-quality speech signal: a low number of artifacts and consistent loudness and intonation are observed. The system was likely trained on clean studio data.

According to the DNSMOS and MOSNet results, both systems are practically on par. The difference is only 0.05 points, which is within the statistical margin of error.

The results indicate that both systems provide an acceptable level of quality for the Kyrgyz language. However, TurkicTTS achieves slightly higher scores than the MMS system across all three metrics.

The difference between the DNSMOS metrics is only 0.05 points; both metrics remain within an acceptable quality range, indicating that the synthetic speech is of sufficient quality.

The MOSNet metric evaluates speech naturalness and prosodic characteristics (TurkicTTS 3.80, MMS 3.75). Although the difference is small, TurkicTTS shows a consistent advantage. These values are lower than those obtained for Uzbek (3.99–

4.04), which is explained by the small amount of available training data for Kyrgyz and its phonetic characteristics.

The most significant advantage was observed in the NISQA metric, with 4.20 for TurkicTTS and 4.00 for MMS. The difference is 0. 20 points. This result is consistent with the trend observed for Uzbek (TurkicTTS – 4.44, MMS – 4.26), indicating that the TurkicTTS architecture consistently generates acoustically cleaner signals.

The fully-processed audio corpora are available at the following repositories: https://huggingface.co/datasets/KazNU-Lab/

Currently, the repositories are private pending copyright registration for the processed audio corpora. This is due to the lack of ready parallel audio corpora that simultaneously cover several Turkic languages in open access. Therefore, the created resource has both scientific and practical value for machine translation, speech synthesis, and multimodal processing.

4. Discussion

The evaluation results show that ChatGPT and DeepSeek achieve the highest performance for Kazakh-Kyrgyz and Kazakh-Uzbek translations according to several automatic indicators (Table 2). Here, ChatGPT shows a good performance in COMET and ChrF results, which indicates a high level of semantic sufficiency and fluency. DeepSeek has similar performance and can be considered a competitive solution. In contrast, NLLB-200-3.3 and Gemma-2-27b achieve slightly lower scores, indicating that although the models can perform the translation task, they do not reach the same level of accuracy and fluency as the leading systems. Despite their good performance, both ChatGPT and DeepSeek are not suitable for large-scale corpus generation in this study due to limited free access and practical limitations in processing large amounts of data. Among the models that can be used via an available API or local deployment, Gemma and NLLB offer realistic options. Gemma showed slightly better translation quality; however, its processing speed was found to be a significant limitation. As shown in Table 3, in order to improve processing speed, the NLLB-200-3.3 model was selected as the preferred option, as it provides a more optimal balance between translation quality and computational efficiency.

Using the NLLB-200-3.3 model, a large-scale machine translation project was conducted from Kazakh into Kyrgyz and Kazakh into Uzbek with the aim of creating a parallel corpus. The overall quality of the translation was generally satisfactory. However, during a systematic manual review of the translated text, a number of recurring errors were identified that require attention.

As a result of analyzing and cleaning the corpus from 300 000 sentences, shown in table 4, 8322–9307 repeated lines were identified and removed (depending on the language pair and corpus version). Removing these repetitions improved the structural and lexical consistency of the training material. Another important source of errors was the incorrect representation of abbreviations, toponyms, and official names. In the context of closely related Turkic languages, the models often replaced country names and official abbreviations with equivalents from another language. For example, cases of replacing the word “Kazakhstan” with “Kyrgyzstan” or “O’zbekiston” were recorded, as well as incorrect interpretation of abbreviations such as “КР” and “ЖШС”.

Comparative analysis showed that in the Kyrgyz part of the corpus, about 11000 out of 300 000 sentences were incorrect, which is about 3.7% of the total data. In the Uzbek part of the corpus, the number of corrections was slightly higher: 12930 examples (9300 lexical repetitions and 3630 incorrect abbreviations), which corresponds to 4.3% of the total number of sentences. These figures indicate the systematic nature of errors that occur during the automatic generation of synthetic data for closely related Turkic languages. Despite the relatively moderate proportion of errors (3.7–4.3%), such errors significantly affect the training process of neural machine translation models, contributing to overfitting, lexical-semantic distortions, and reduced generalization capabilities. Pre-cleaning improved the structural consistency of the corpus.

Despite these shortcomings, NLLB-200-3.3 demonstrates considerable promise for generating parallel data for low-resource Turkic language pairs. However, the findings confirm that automatic translation alone cannot guarantee sufficient corpus quality, and post-editing remains an essential step – particularly for handling abbreviations, named entities, domain-specific terminology, and complex syntactic constructions. Furthermore, fine-tuning the model on Kazakh–Kyrgyz and Kazakh–Uzbek specific data is expected to enhance its accuracy and reliability in future applications.

Following the completion of the initial manual review phase, the most obvious errors—such as in-

correct abbreviations, word repetitions, and clear syntactic and semantic inconsistencies–were corrected. This led to a significant improvement in the overall quality of the corpus. However, this phase should be considered preliminary.

The results of objective evaluations of synthetic speech quality in Kyrgyz and Uzbek demonstrate that the proposed Text-based audio synthesis methodology applies to both languages. Moreover, the results indicate that resource disparities between the languages directly affect synthesis quality. Overall, a similar trend was observed across both languages: while the TurkicTTS system consistently performed better on the NISQA metric, the results of the two systems were close for the DNSMOS and MOSNet scores. This indicates that the TurkicTTS system is inclined to generate a speech signal with high acoustic quality and few artifacts, and that, in terms of overall naturalness and perceptual quality, the MMS-TTS and TurkicTTS systems are comparable.

The results obtained for Uzbek demonstrate that synthetic speech was generated at a sufficiently high quality. In particular, the NISQA metrics scores ranging from 4.26 to 4.44 indicate that the synthesized speech is close to natural pronunciation. This demonstrates that the TTS systems used for Uzbek are suitable for creating a parallel audio corpus. Moreover, the systems' advantages are evident across various aspects: While TurkicTTS excels in acoustic clarity and signal quality, MMS-TTS achieved a higher SSL-MOS score, which evaluates prosodic naturalness. For example, the SSL-MOS score was 3.24 for MMS-TTS and 2.39 for TurkicTTS. This suggests that the MMS-TTS system may better capture the naturalness of intonation, rhythm, and speech flow.

Although the quality metrics obtained for Kyrgyz were lower than those for Uzbek, they still demonstrated quality close to natural speech. The low MOSNet, DNSMOS, and NISQA scores are explained by the limited availability of Kyrgyz speech resources, insufficient phonetic coverage in multilingual models, and the absence of dedicated Kyrgyz TTS systems.

Overall, TTS results demonstrate that the quality of synthetic speech is directly dependent on the level of a language's resource availability. Therefore, expanding the textual and speech corpora in Kyrgyz, improving phonetic normalization, and adapting TTS models remain important directions for future research. The proposed Text-based audio synthesis methodology provides a practical basis for creating parallel audio corpora for low-resource Turkic languages.

5. Conclusions

This study presented an AI-driven methodology for constructing parallel text corpora and synthesizing audio datasets for the Kyrgyz–Kazakh and Uzbek–Kazakh language pairs, addressing the critical shortage of high-quality linguistic and speech resources for these low-resource Turkic languages.

The experimental results demonstrated that while ChatGPT and DeepSeek achieved the highest translation quality, their practical limitations in processing large volumes of data made them unsuitable for large-scale corpus generation. Consequently, NLLB-200-3.3 was selected as the most viable option, offering a balanced trade-off between translation quality and computational efficiency. The large-scale translation of 300,000 Kazakh sentences into Kyrgyz and Uzbek revealed systematic errors affecting approximately 3.7% and 4.3% of the data respectively, primarily involving incorrect abbreviations, country name substitutions, and lexical repetitions. Although these error rates are relatively moderate, their impact on neural model training is significant, contributing to overfitting and lexical-semantic distortions.

With regard to audio synthesis, the evaluation results confirmed that the proposed text-based audio synthesis methodology is applicable to both Kyrgyz and Uzbek. The TurkicTTS system consistently demonstrated superior acoustic quality, while MMS-TTS showed advantages in prosodic naturalness, particularly for Uzbek. The lower synthesis quality observed for Kyrgyz is directly attributable to the limited availability of speech resources and the absence of dedicated Kyrgyz TTS systems, highlighting the need for expanded phonetic corpora and language-specific model adaptation as priorities for future research.

Given the rapid development of AI models and tools, there is room for further improvement of the obtained results and improvement of the proposed methodology. Future research will be aimed at fully correcting the identified errors and increasing the quality of the data.

The constructed Kyrgyz-Kazakh and Kazakh-Uzbek parallel corpora will serve as a basis for training "lightweight" LLM models that can operate on low computational resources. In addition,

the proposed methodology can be used to build parallel corpora for other Turkic languages and develop high-quality neural machine translation systems for resource-constrained Turkic languages.

Funding

This work was supported by the Ministry of Science and Higher Education of the Republic of Kazakhstan under the grant project «Study of automatic generation of parallel speech corpora of Turkic languages and their use for neural models» (grant number IRN AP23488624).

Conflicts of Interest

The authors declare no conflict of interest.

Author Contributions

Conceptualization, B.A.; methodology, A.K.; software, A.K. and B.A.; validation, B.A., and A.K.; formal analysis, B.A.; investigation, A.K.; resources, B.A. and A.K; data curation, A.K.; writing—original draft preparation, A.K. and B.A.; writing–review and editing, B.A. and A.K.; visualization, B.A.; supervision, A.K.; project administration, A.K. All authors have read and agreed to the published version of the manuscript.

References

1. Ethnologue. How Many Languages Are There in the World? Available online: https://www.ethnologue.com/insights/how-many-languages/ (accessed on 30 July 2025).
2. Cieri, C.; Maxwell, M.; Strassel, S.; Tracey, J. Selection Criteria for Low-Resource Language Programs. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation* (LREC'16), Portorož, Slovenia, 2016; European Language Resources Association (ELRA), pp. 4543–4549.
3. Ivasubramanian, R.; Umamaheswari, T.; Babu, S.B.G.; Inakoti, R.; Salome, J.; Y.M., Dr; Sivasubramanian, R. Natural Language Processing in Low-Resource Language Contexts. *Front. Health Inform*. 2024, 13, 1578–1584. doi:10.52783/fhi.vi.1776.
4. Pakray, P.; Gelbukh, A.; Bandyopadhyay, S. Natural language processing applications for low-resource languages. *Nat. Lang. Process*. 2025, 31, 183–197. doi:10.1017/nlp.2024.33.
5. Bekarystankyzy, A.; Mamyrbayev, O.; Mendes, M.; Fazylzhanova, A.; Assam, M. Multilingual end-to-end ASR for low-resource Turkic languages with common alphabets. *Sci. Rep*. 2024, 14, 10.1038/s41598-024-64848-1.
6. Tukeyev, U.; Amirova, D.; Karibayeva, A.; Sundetova, A.; Abduali, B. Combined Technology of Lexical Selection in Rule-Based Machine Translation. In *Recent Advances in Systems, Control and Information Technology*; Springer: Cham, Switzerland, 2017; pp. 491–500. doi:10.1007/978-3-319-67077-5_47.
7. Tukeyev, U.; Karibayeva, A.; Abduali, B. Neural Machine Translation System for the Kazakh Language Based on Synthetic Corpora. *MATEC Web Conf*. 2019, 252, 03006. doi:10.1051/matecconf/201925203006.
8. Ahmadnia, B.; Serrano, J.; Haffari, G. Persian-Spanish Low-Resource Statistical Machine Translation Through English as Pivot Language. In *Proceedings of Recent Advances in Natural Language Processing*, RANLP 2017, Varna, Bulgaria, 2017; INCOMA Ltd, pp. 24–30.
9. Elmadani, K.N.; Buys, J. Neural Machine Translation between Low-Resource Languages with Synthetic Pivoting. In *Proceedings of LREC-COLING 2024*, Torino, Italia, 2024; ELRA and ICCL, pp. 12144–12158.
10. Pontes, J.J.A. Bilingual Sentence Alignment of a Parallel Corpus by Using English as a Pivot Language. In *Proceedings of JISIC*, Quito, Ecuador, 2014; Association for Computational Linguistics, pp. 13–20.
11. Alekseev, A.; Turatali, T. KyrgyzNLP: Challenges, Progress, and Future. arXiv 2024, arXiv:2411.05503v1.
12. Riemland, M. Theorizing Sustainable, Low-Resource MT in Development Settings: Pivot-Based MT between Guatemala's Indigenous Mayan Languages. *Transl. Spaces* 2023, 12, doi:10.1075/ts.22018.rie.
13. Lin, D.; Murakami, Y.; Ishida, T. Towards Language Service Creation and Customization for Low-Resource Languages. *Information* 2020, 11, 67. doi:10.3390/info11020067.
14. Trieu, H.-L.; Tran, V.; Ittoo, A.; Nguyen, L. Leveraging Additional Resources for Improving Statistical Machine Translation on Asian Low-Resource Languages. ACM Trans. Asian Low-Resour. Lang. *Inf. Process*. 2019, 18, 1–22. doi:10.1145/3314936.
15. Tiedemann, J. Parallel Data, Tools and Interfaces in OPUS. In *Proceedings of the 8th International Conference on Language Resources and Evaluation*, LREC 2012, pp. 2214–2218.
16. Karakanta, A.; Orrego-Carmona, D. Subtitling in Transition: The Case of TED Talks. 2023, doi:10.1075/ata.xx.07kar.
17. Koehn, P. Europarl: A Parallel Corpus for Statistical Machine Translation. In *Proceedings of Machine Translation Summit X*, Phuket, Thailand, 2005; pp. 79–86.
18. Tiedemann, J.; Thottingal, S. OPUS-MT—Building Open Translation Services for the World. In *Proceedings of the 22nd Annual Conference of the European Association for Machine Translation*, 2020.
19. Balahur, A.; Turchi, M. Comparative Experiments Using Supervised Learning and Machine Translation for Multilingual Sentiment Analysis. *Comput. Speech Lang*. 2014, 28, 56–75. doi:10.1016/j.csl.2013.03.004.
20. Bahdanau, D.; Cho, K.; Bengio, Y. Neural Machine Translation by Jointly Learning to Align and Translate. arXiv 2015, arXiv:1409.0473.

21. Koehn, P.; Knowles, R. Six Challenges for Neural Machine Translation. In *Proceedings of the First Workshop on Neural Machine Translation*, 2020; pp. 28–39.

22. Conneau, A.; Khandelwal, K.; Goyal, N.; Chaudhary, V.; Wenzek, G.; Guzmán, F.; Grave, E.; Ott, M.; Zettlemoyer, L.; Stoyanov, V. Unsupervised Cross-lingual Representation Learning at Scale. In *Proceedings of the 58th ACL*, 2020; pp. 8440–8451.

23. Costa-jussà, M.R.; Cross, J.; et al. No Language Left Behind: Scaling Human-Centered Machine Translation. arXiv 2022, arXiv:2207.04672.

24. Hou, X. Research on Translation Corpus Building with the Assistance of AI. In *Proceedings of the 2021 International Conference on Smart Technologies and Systems for IoT*, 2022; pp. 136–140.

25. Reixa, I.D.; et al. Corpus PaGeS: A Multifunctional Resource for Language Learning, Translation and Cross-Linguistic Research. In *Parallel Corpus for Contrastive and Translation Studies*; John Benjamins: Amsterdam, Netherlands, 2019; pp. 103–121.

26. Post, M. A Call for Clarity in Reporting BLEU Scores. In *Proceedings of the Third Conference on Machine Translation*, 2018; pp. 186–191.

27. Abduali, B.; Tukeyev, U.; Zhumanov, Z.; Israilova, N. Study of Kyrgyz-Kazakh Neural Machine Translation. In *Recent Challenges in Intelligent Information and Database Systems*; Nguyen, N.T., et al., Eds.; Springer: Singapore, 2025; CCIS, Vol. 2493, pp. . doi:10.1007/978-981-96-5881-7_21.

28. Zhumanov, Z.; et al. Integrated Technology for Creating Quality Parallel Corpus. In *Advances in Computational Collective Intelligence*; pp. 511–524.

29. Papineni, K.; Roukos, S.; Ward, T.; Zhu, W.J. BLEU: A Method for Automatic Evaluation of Machine Translation. In *Proceedings of ACL 2002*, 2002.

30. Popović, M. chrF: Character n-gram F-score for Automatic MT Evaluation. In *Proceedings of WMT 2015*, 2015.

31. Rei, R.; Sánchez-Cartagena, V.; Popović, M. COMET: A Neural Framework for MT Evaluation. In *Proceedings of EMNLP 2020*, 2020.

32. Wu, Y.; Schuster, M.; Chen, Z.; Le, Q.V.; Norouzi, M.; Macherey, W.; Krikun, M.; Cao, Y.; Gao, Q.; Macherey, K.; et al. Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation. arXiv 2016, arXiv:1609.08144.

33. Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. Language Models are Few-Shot Learners. arXiv 2020, arXiv:2005.14165.

34. Costa-jussà, M.R.; Cross, J.; Fan, A.; Ghazvininejad, M.; Gu, J.; Gunasekara, C.; He, Y.; Kalbassi, E.; Liptchinsky, V.; Liu, Z.; et al. No Language Left Behind: Scaling Human-Centered Machine Translation. arXiv 2022, arXiv:2207.04672.

35. DeepSeek-AI, Guo, D., Yang, D., Zhang, H., Song, J., Wang, P., … & Zhang, Z. (2025). *DeepSeek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning*. arXiv. https://arxiv.org/abs/2501.12948

36. Ziegler, A.; Kalliamvakou, E.; Li, X.A.; Rice, A.; Rifkin, D.; Simister, S.; Sittampalam, G.; Aftandilian, E. Measuring GitHub Copilot's Impact on Productivity. *Commun*. ACM 2024, 67, 54–63. doi:10.1145/3633453.

37. Gemma Team; Mesnard, T.; et al. Gemma: Open Models Based on Gemini Research and Technology. arXiv 2024, arXiv:2403.08295.

Information about the authors:

Aidana Karibayeva – PhD, is an Acting Associate Professor at the Department of Information Systems of al-Farabi Kazakh National University. Dr. Karibayeva has more than 13 years of experience in artificial intelligence and machine learning. Her research interests include neural networks, deep learning applications, data mining, and speech recognition (Almaty, Kazakhstan, e-mail: karibayeva.aidana@kaznu.edu.kz. ORCID iD: 0000-0002-2023-1573).

Balzhan Abduali – Master of Engineering Science, Senior Lecturer at the Department of Artificial Intelligence and Big Data of al-Farabi Kazakh National University. She has more than 12 years of experience in machine learning. Her research interests include artificial intelligence, machine learning, and data mining (Almaty, Kazakhstan, e-mail: abduali.balzhan@kaznu.kz. ORCID iD: 0000-0003-0140-4181).

Submission received: 8 May, 2026
Revised: 25 May, 2026
Accepted: 25 May, 2026

IRSTI 28.23.37; 49.33.29 https://doi.org/10.26577/jpcsit4220269

Yedil Nurakhov[1], Abzal Kyzyrkanov[2*], Syrym Aldabergen[3]

[1]Al-Farabi Kazakh National University, Almaty, Kazakhstan
[2]Astana IT University, Astana, Kazakhstan
[3]JSC "Digital Development Center of the National Bank of Kazakhstan", Almaty, Kazakhstan
*e-mail: abzzall@gmail.com

PPO-BASED PHYSICAL RESOURCE BLOCK SCHEDULING IN 5G NR: A REINFORCEMENT LEARNING APPROACH WITH 3GPP-COMPLIANT CHANNEL MODELING

Abstract. This paper proposes a deep reinforcement learning approach to Physical Resource Block (PRB) scheduling in 5G New Radio networks. A Gymnasium-compatible simulation environment is developed on the 3GPP TR 38.901 UMa NLOS channel model with Rayleigh fading, serving as a reproducible experimental platform. A Proximal Policy Optimization (PPO) agent is trained with a composite reward function that jointly optimizes aggregate throughput and Jain's Fairness Index. The agent is evaluated over 100 episodes with five independent random seeds against three classical baselines – Round Robin, Proportional Fair, and Maximum CQI. The PPO agent achieves a mean throughput of 57.09 ± 1.04 Mbps with a Jain's Fairness Index of 0.358, surpassing Round Robin (32.37 Mbps, JFI 0.630) and Proportional Fair (35.31 Mbps, JFI 0.597) in throughput while exceeding Maximum CQI in fairness (JFI 0.100). A throughput-fairness tradeoff analysis confirms that the proposed agent occupies an operating point inaccessible to any single classical method. The composite reward function, which weights both objectives equally, enables learned adaptive behavior that balances spectral efficiency and user equity. The code and evaluation pipeline are fully open-source and reproducible using the Stable-Baselines3 library.

Keywords: 5G NR, physical resource block scheduling, reinforcement learning, proximal policy optimization, Jain's fairness index, 3GPP channel model, radio resource management.

1. Introduction

The exponential growth of mobile data traffic in fifth generation (5G) networks has placed unprecedented demands on radio resource management systems. According to Cisco's Annual Internet Report, global mobile data traffic is projected to grow threefold between 2020 and 2025, driven by video streaming, IoT deployments, and emerging applications such as augmented reality [1]. The International Telecommunication Union similarly projects that IMT traffic will increase by a factor of ten by 2030 [2]. In this context, the efficient allocation of Physical Resource Blocks (PRB) at the base station scheduler becomes a critical bottleneck determining overall network performance.

The PRB scheduling problem can be stated as follows. A base station operating under the 5G New Radio (NR) standard [5] possesses a finite pool of PRBs which must be allocated among active User Equipment (UE) at every Transmission Time Interval (TTI) of one millisecond. The channel quality experienced by each UE, quantified through the Channel Quality Indicator (CQI), varies continuously due to path loss, shadowing, and fast fading [14]. A scheduler must therefore make sequential allocation decisions under uncertainty, simultaneously maximizing aggregate throughput and ensuring fair resource distribution among users – two objectives that are fundamentally in tension [3], [4]. The combinatorial nature of this decision problem, combined with the non-stationary channel environment, renders optimal closed-form solutions intractable for practical network deployments [5].

Classical scheduling algorithms address this problem through fixed heuristic rules. Round Robin (RR) allocates resources in a cyclic fashion, guaranteeing fairness at the cost of throughput efficiency [3]. The Proportional Fair (PF) scheduler, introduced by Kelly [4] and widely adopted in 4G and 5G deployments, strikes a balance between throughput and fairness by prioritizing users whose

instantaneous rate exceeds their long-term average. The Maximum CQI scheduler greedily allocates all resources to the user with the best channel, maximizing spectral efficiency while completely disregarding fairness. While these methods are computationally efficient and well understood, they rely on fixed priority rules that cannot adapt to the dynamic statistical structure of real traffic and channel conditions [6], [13].

The emergence of deep reinforcement learning (DRL) has opened new avenues for data-driven radio resource management. Deep Q-Networks (DQN), introduced by Mnih et al. [7], demonstrated that neural networks could learn near-optimal policies for sequential decision problems directly from interaction with an environment. Subsequent work applied DQN to distributed spectrum access [8] and multi-user resource allocation, showing improvements over classical baselines in dynamic environments. The Proximal Policy Optimization (PPO) algorithm [9], an on-policy actor-critic method, further improved training stability through a clipped surrogate objective, making it particularly suitable for environments with large discrete action spaces such as PRB allocation. Gu et al. [10] demonstrated the applicability of knowledge-assisted DRL to 5G scheduler design, bridging theoretical frameworks and practical implementation. Broader surveys confirm that DRL-based approaches consistently outperform classical methods across a range of resource management tasks in heterogeneous wireless networks [11], [12].

Despite this progress, several important gaps remain in existing literature. First, many DRL-based scheduling studies rely on simplified channel models that do not conform to 3GPP specifications, limiting the practical relevance of reported results [13], [14]. Second, most works optimize either throughput or fairness in isolation, whereas real network operators require schedulers that jointly optimize both objectives under a unified reward signal. Third, systematic benchmarking against multiple classical baselines within a single reproducible experimental environment remains uncommon, making it difficult to assess the true advantage of DRL approaches [13]. The work of Sun et al. [12] and Alwarafy et al. [11] identify these shortcomings as open research directions, yet they have not been comprehensively addressed in the context of PRB scheduling under 3GPP-compliant channel conditions.

The central research question of this paper is therefore: *can a PPO-based agent learn a scheduling policy that simultaneously outperforms Round Robin, Proportional Fair, and Maximum CQI baselines in terms of aggregate throughput and Jain's Fairness Index, when trained and evaluated in a simulation environment built on the 3GPP TR 38.901 UMa NLOS channel model?*

The contributions of this paper are as follows. First, we develop a Gymnasium-compatible simulation environment for PRB scheduling that implements the 3GPP UMa NLOS channel model [14], providing a reproducible and standards-compliant experimental platform. Second, we propose a PPO-based scheduling agent with a composite reward function that jointly optimizes throughput and Jain's Fairness Index through a weighted combination. Third, we conduct a systematic evaluation against three classical baselines Round Robin, Proportional Fair, and Maximum CQI – across 100 evaluation episodes with five independent random seeds, reporting means and 95% confidence intervals. Fourth, we demonstrate through a throughput-fairness tradeoff analysis that the proposed agent achieves an operating point inaccessible to any single classical method, confirming the value of learned adaptive policies for this problem. The experimental results are fully reproducible using the open-source code accompanying this paper, implemented with the Stable-Baselines3 library [15].

2. Materials and methods

2.1 System model and Problem formulation

We consider a single cell 5G NR downlink scenario with one gNodeB (gNB) serving N = 10 User Equipment (UE) devices simultaneously. The gNB allocates a pool of K = 50 Physical Resource Blocks per Transmission Time Interval (TTI) of 1 ms. Each PRB occupies a bandwidth of 180 kHz (15 kHz subcarrier spacing, 12 subcarriers). The scheduling decision is made at every TTI based on the observed channel state and buffer occupancy of each UE.

The instantaneous throughput achievable by UE i when allocated k_i PRBs is given by the Shannon capacity formula:

$$R_i = k_i * B_{PRB} * \log_2(1 + SNR_i) \quad (1)$$

where B_{PRB} = 180 kHz is the PRB bandwidth and SNR_i is the signal-to-noise ratio of UE i, derived from its CQI report via the 3GPP CQI-to-SNR mapping table [14].

The scheduling problem is formulated as a Markov Decision Process (MDP) defined by the tuple (S, A, P, R, γ), where S is the state space, A is the action space, P is the transition probability function, R is the reward function, and γ = 0.99 is the discount factor. The agent interacts with the environment over episodes of T = 1000 steps each.

2.2 Channel model

The channel model implements the 3GPP TR 38.901 Urban Macro (UMa) NLOS path loss formula at a carrier frequency of 3.5 GHz:

$$PL = 36.7 * \log_{10}(d) + \\ +22.7 + 26 * \log_{10}(f_{GHz}) \ [dB] \quad (2)$$

where d ∈ [50, 2000] is the UE distance from the gNB and f_{GHz}= 3.5. Three propagation components are combined:

(i) UMa NLOS path loss per Equation (2);

(ii) log-normal shadowing with zero mean and standard deviation of 8 dB;

(iii) Rayleigh fast fading, modeled as a circularly symmetric complex Gaussian channel with unit variance. The resulting SNR is mapped to a CQI value in [1, 15] using the 3GPP TS 36.213 CQI table [14]. The UE distances are drawn uniformly at the start of each episode and held fixed; shadowing is a persistent per-episode random variable, while Rayleigh fading changes at every TTI, providing the non-stationary dynamics that motivate adaptive scheduling.

2.3 State, action and reward

State space S is a 30-dimensional normalized observation vector composed of three 10-dimensional blocks per UE:

$$s_t = \left[\frac{CQI_i}{15}, \frac{buf_i}{buf_{max}}, \frac{\widetilde{R}_i^{EMA}}{tp_{norm}}\right]_{i=1}^{10} \in [0,1]^{30} \quad (3)$$

where $CQI_i \in \{1, \cdots, 15\}$ is the current channel quality indicator, $buf_i \in [0, 1000]$ packets is the buffer occupancy, and $\widetilde{R}_i^{EMA}(t)$ is an exponential moving average of UE i's delivered throughput with smoothing coefficient α = 0.05:

$$\widetilde{R}_i^{EMA} = (1-\alpha)\,\widetilde{R}_i^{EMA}(t-1) + \alpha R_i(t) \quad (3a)$$

The EMA component provides the agent with a memory of historical channel utilization, analogous to the long-term throughput estimate used by the Proportional Fair scheduler.

Action space A = $[0,1]^{10}$ is a continuous vector of allocation weights $w = (w_1, \cdots, w_N)$. The number of PRBs assigned to UE i are computed as:

$$k_i = \left[\frac{w_i}{\sum_{j=1}^{N} w_j} * K\right] \quad (4)$$

where $K = 50$ is the total number of available PRBs and $\lfloor\cdot\rceil$denotes rounding to the nearest integer. This continuous parameterization avoids the combinatorial explosion of a discrete MultiDiscrete $([51]^{10}) \approx 10^{17}$ action space, enabling stable learning with PPO.

The scalar reward at each TTI is a convex combination of normalized throughput and Jain's Fairness Index:

$$r_t = 0.5 * \frac{R_{total}}{R_{max}} + 0.5 * J(R_1, \cdots, R_N) \quad (5)$$

where $R_{\text{total}} = \sum_{i=1}^{N} R_i$ is the aggregate throughput, $R_{\max}$ is the theoretical maximum throughput at $\text{CQI} = 15$for all PRBs, and Jain's Fairness Index is defined as:

$$J(R_1, \cdots, R_N) = \frac{\left(\sum_{i=1}^{N} R_i\right)^2}{N * \sum_{i=1}^{N} R_i^2} \in \left[\frac{1}{N}, 1\right] \quad (6)$$

A value of J = 1 corresponds to perfectly equal resource distribution, while J = 1/N indicates that all resources are concentrated at a single user. The equal weighting of throughput and fairness in Equation (5) reflects a neutral policy that does not privilege either objective.

2.4 PPO agent and Training procedure

The PPO agent is implemented using the Stable-Baselines3 library [15] with an MlpPolicy consisting of two hidden layers of 128 units each. The actor and critic share this architecture. Key hyperparameters are listed in Table 1.

Table 1
PPO Hyperparameters

Hyperparameter	Value
Total timesteps	1,000,000
Parallel environments	4
Network architecture	[128, 128] MLP
Learning rate (initial)	3×10^{-4} (linear decay)
Rollout length (n_steps)	2048
Mini-batch size	512
Optimization epochs	10
Discount factor (γ)	0.99
GAE lambda (λ)	0.95
Clipping parameter (ε)	0.2
Entropy coefficient	0.005

Training employs a linearly decaying learning rate schedule from 3×10^{-4} to a floor of 5% of the initial value, preventing entropy collapse in late-stage training. The agent is trained independently for five random seeds (0–4) to assess run-to-run variability. A checkpoint is saved every 100,000 environment steps, and an EvalCallback monitors performance on a held-out evaluation environment (seeded with seed + 100) every 100,000 steps, retaining the best-performing checkpoint.

2.5 Baseline algorithms

Three deterministic baseline schedulers are implemented for comparison:

Round Robin (RR): Allocates resources in a strictly cyclic order, assigning $k_i = \frac{K}{N}$ PRBs to each UE at every TTI regardless of channel quality or buffer state. This guarantees temporal fairness but ignores channel diversity.

Maximum CQI (Max-CQI): Allocates all K PRBs to the UE with the highest instantaneous CQI at each TTI. This is the greedy spectral efficiency maximizer, yielding maximum aggregate throughput but concentrating resources on the best-channel user.

Proportional Fair (PF): At each TTI, UE i is prioritized according to the ratio $\frac{CQI_i}{\bar{R}_i^{EMA}}$ where $\bar{R}_i^{EMA}$ is an exponential moving average of received throughput. PF balances instantaneous channel quality against historical fairness and is the industry-standard scheduler in 4G/5G systems.

2.6 Evaluation protocol

Each method is evaluated over 100 episodes of 1000 TTIs each, with five independent random seeds for the PPO agent. The following metrics are recorded per episode:

(i) mean aggregate throughput in Mbps;
(ii) Jain's Fairness Index;
(iii) composite reward;
(iv) mean latency in ms.

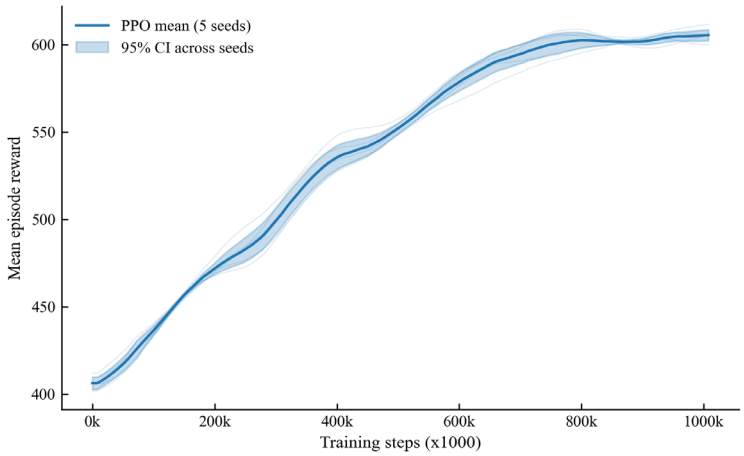
Results are reported as means with 95% confidence intervals (CI) computed as ±1.96 × SEM, where SEM is the standard error of the mean across 100 episodes.

3. Results

3.1 Learning dynamics

Figure 1 shows the learning curve of the PPO agent over 1,000,000 training timesteps, averaged across five seeds with shaded 95% confidence bands. The agent exhibits rapid initial improvement in the first 200,000 steps as it learns to exploit channel quality information from the CQI components of the observation. The reward stabilizes between 400,000 and 600,000 steps, reaching a plateau of approximately 0.61 ± 0.02 on the composite reward scale. Convergence is consistent across seeds, with narrow confidence intervals in the steady-state phase, indicating robust learning despite the stochastic channel environment.

Figure 1
PPO training learning curve averaged over 5 seeds (shaded area: 95% confidence interval)



3.2 Comparative performance

Table 2 reports the aggregate performance of PPO and the three baseline schedulers over 100 evaluation episodes. The PPO agent achieves a mean throughput of 57.09 Mbps (95% CI: ±1.04), which represents a 76.4% improvement over Round Robin (32.37 Mbps) and a 61.7% improvement over Proportional Fair (35.31 Mbps). The Maximum CQI baseline achieves the highest raw throughput at 65.25 Mbps, as expected from a pure spectral efficiency maximizer, but at the cost of a Jain's Fairness Index of exactly 0.100 the theoretical minimum for 10 UEs indicating that all resources are monopolized by the strongest UE in every TTI.

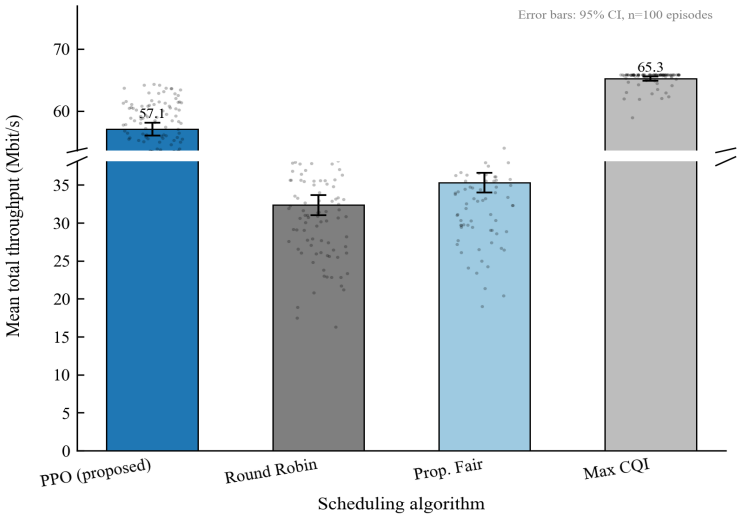
The PPO agent's Jain's Fairness Index of 0.358 significantly exceeds that of Max CQI (0.100, improvement of 3.58×) while remaining below Round Robin (0.630) and Proportional Fair (0.597). This pattern reflects the PPO agent's learned trade-off: it sacrifices some fairness relative to the cyclic schedulers to exploit channel diversity, while providing substantially better fairness guarantees than the greedy Max CQI policy. The composite reward (Equation 5) of 0.612 for PPO is the highest among all methods, confirming that the learned policy most effectively optimizes the joint objective defined during training.

Figure 2 presents box plots of per-episode throughput and fairness distributions across all methods. The narrow interquartile range of Max CQI confirms its deterministic channel-exploitation behavior. The PPO agent shows moderate variability consistently with adapting to different per-episode channel realizations.

Table 2
Performance comparison across schedulers (100 episodes, mean ± 95% CI)

Scheduler	Throughput (Mbps)	Jain's FI	Spectral Eff. (bps/Hz)	Latency (ms)
PPO (ours)	57.09 ± 1.04	0.358 ± 0.011	6.34	191.3 ± 5.4
Round Robin	32.37 ± 1.33	0.630 ± 0.017	3.60	139.5 ± 7.6
Proportional Fair	35.31 ± 1.32	0.597 ± 0.017	3.92	129.5 ± 8.0
Max CQI	65.25 ± 0.34	0.100 ± 0.000	7.25	193.7 ± 6.96

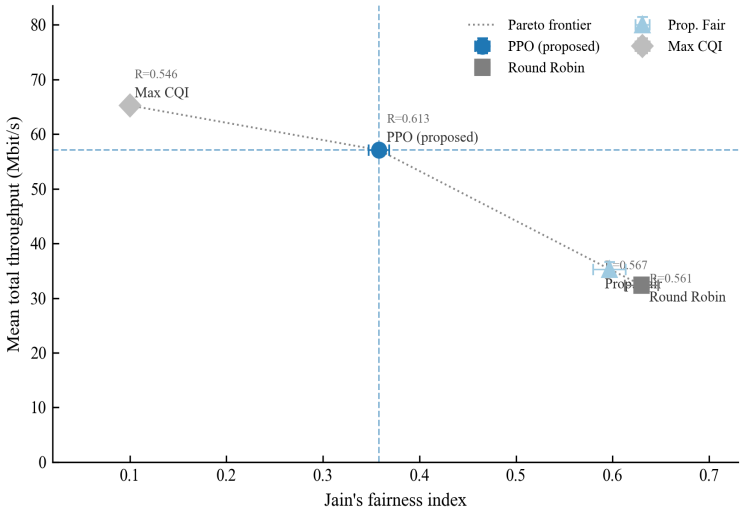
Figure 2
Throughput and fairness comparison across all scheduling methods (100 episodes each). Box plots show median, interquartile range, and whiskers at 5th/95th percentiles



3.3 Throughput-Fairness Tradeoff

Figure 3 plots the mean episode throughput against Jain's Fairness Index for all four schedulers. The classical baselines occupy three corners of the tradeoff space: Round Robin (low throughput, high fairness), Max CQI (high throughput, minimal fairness), and Proportional Fair (intermediate on both axes). The PPO agent occupies a distinct operating point at high throughput and moderate fairness, a region that is inaccessible to any single classical scheduler. Specifically, no classical method simultaneously achieves throughput above 50 Mbps and fairness above 0.30; the PPO agent achieves 57.09 Mbps at 0.358.

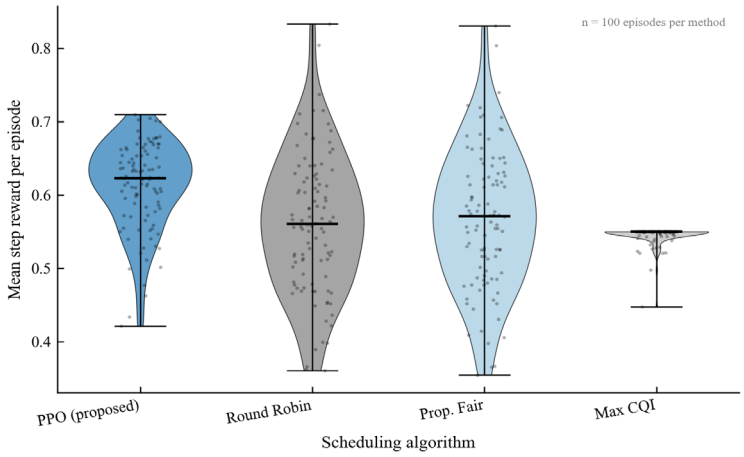
Figure 3
Throughput-fairness tradeoff. Each point represents the mean over 100 episodes. The PPO agent occupies a region unreachable by any individual classical baseline, demonstrating the advantage of learned adaptive scheduling



3.4 Reward distribution

Figure 4 shows the distribution of composite reward values across episodes for all methods. The PPO agent achieves the highest median reward (0.623) with a relatively compact distribution, demonstrating consistent joint optimization of throughput and fairness. Proportional Fair and Round Robin achieve comparable median rewards (0.571 and 0.560, respectively) but with substantially wider variance. Max CQI exhibits the narrowest distribution due to its deterministic greedy behavior but converges to a lower median of 0.550 because the fairness penalty dominates any gains in throughput.

Figure 4
Composite reward distribution across 100 evaluation episodes per scheduler. The dashed line indicates the PPO median (0.623)



The tighter reward distribution of PPO relative to Round Robin and Proportional Fair (standard deviation 0.060 vs. 0.094 and 0.094, respectively) indicates that the learned policy is more consistent across varying channel realizations, an important property for network operators who require predictable quality of service.

4. Conclusion

This paper presented a PPO-based approach to Physical Resource Block scheduling in 5G NR networks. A Gymnasium-compatible simulation environment conforming to the 3GPP TR 38.901 UMa NLOS channel model was developed, providing a reproducible experimental platform with realistic propagation characteristics including path loss, log-normal shadowing, and Rayleigh fading. A PPO agent was trained with a composite reward function that jointly optimizes aggregate throughput and Jain's Fairness Index through equal weighting, enabling the agent to learn adaptive allocation policies that balance spectral efficiency with user equity.

Experimental evaluation over 100 episodes with five independent random seeds demonstrated that the PPO agent achieves a mean throughput of 57.09 Mbps a 76.4% improvement over Round Robin and 61.7% improvement over Proportional Fair while maintaining a Jain's Fairness Index of 0.358, which is 3.58 times higher than Maximum CQI. The throughput-fairness tradeoff analysis confirmed that the PPO agent occupies an operating point inaccessible to any single classical scheduler, validating the hypothesis that learned adaptive policies can effectively navigate the inherent tension between throughput maximization and fair resource distribution.

The PPO agent's composite reward of 0.612 exceeded all three classical baselines under the defined joint objective, with lower episode-to-episode variance than heuristic methods, suggesting consistent performance across diverse channel realizations. These results support the broader applicability of DRL-based scheduling in heterogeneous wireless network environments.

Several directions remain for future work. First, extending the environment to multi-cell scenarios

with inter-cell interference would increase practical relevance. Second, incorporating traffic heterogeneity including mixed delay-sensitive and best-effort flows would test the scheduler's ability to handle quality-of-service differentiation. Third, exploration of offline RL approaches trained on logged network traces could reduce the sim-to-real gap. Finally, the sensitivity of the learned policy to reward weighting between throughput and fairness merits systematic ablation, as different operator priorities may require different composite objective formulations.

Funding

This research was funded by the Committee of Science of the Ministry of Science and Higher Education of the Republic of Kazakhstan (Grant No. BR24993211).

Conflicts of Interest

The authors declare no conflict of interest.

Author Contributions

Conceptualization, Y.N. and A.K.; Methodology, A.K.; Software, Y.N. and A.S.; Validation, Y.N. and A.K.; Formal Analysis, Y.N.; Investigation, Y.N. and A.S.; Resources, A.K. and A.S.; Data Curation, Y.N.; Writing – Original Draft Preparation, Y.N.; Writing – Review & Editing, A.K.; Visualization, Y.N. and A.S.; Supervision, A.K.; Project Administration, A.K.; Funding Acquisition, A.K.

References

1. Cisco Systems, "Cisco annual internet report (2018–2023) white paper," Cisco, San Jose, CA, USA, Rep., Mar. 2020. [Online]. Available: https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-741490.html

2. ITU-R, "IMT traffic estimates for the years 2020 to 2030," International Telecommunication Union, Geneva, Switzerland, Rep. ITU-R M.2370-0, Jul. 2015.

3. A. Jalali, R. Padovani, and R. Pankaj, "Data throughput of CDMA-HDR a high efficiency-high data rate personal communication wireless system," in *Proc. IEEE 51st Veh. Technol. Conf. (VTC Spring)*, Tokyo, Japan, May 2000, pp. 1854–1858.

4. F. Kelly, "Charging and rate control for elastic traffic," *Eur. Trans. Telecommun.*, vol. 8, no. 1, pp. 33–37, Jan. 1997.

5. 3GPP, "NR; Physical layer procedures for data," 3rd Generation Partnership Project, Sophia Antipolis, France, Tech. Spec. TS 38.214, ver. 17.4.0, Dec. 2022.

6. C. Jiang, H. Zhang, Y. Ren, Z. Han, K.-C. Chen, and L. Hanzo, "Machine learning paradigms for next-generation wireless networks," *IEEE Wireless Commun.*, vol. 24, no. 2, pp. 98–105, Apr. 2017.

7. V. Mnih et al., "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, Feb. 2015.

8. O. Naparstek and K. Cohen, "Deep multi-user reinforcement learning for distributed dynamic spectrum access," *IEEE Trans. Wireless Commun.*, vol. 18, no. 1, pp. 310–323, Jan. 2019.

9. J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," arXiv:1707.06347 [cs.LG], Jul. 2017.

10. Z. Gu, C. She, W. Hardjawana, S. Lumb, D. McKechnie, T. Essery, and B. Vucetic, "Knowledge-assisted deep reinforcement learning in 5G scheduler design: From theoretical framework to implementation," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 7, pp. 2014–2028, Jul. 2021.

11. A. Alwarafy, M. Abdallah, B. S. Ciftler, A. Al-Fuqaha, and M. Hamdi, "Deep reinforcement learning for radio resource allocation and management in next generation heterogeneous wireless networks: A survey," arXiv:2106.00574 [cs.NI], Jun. 2021.

12. Y. Sun, M. Peng, Y. Zhou, Y. Huang, and S. Mao, "Application of machine learning in wireless networks: Key techniques and open issues," IEEE Commun. Surveys Tuts., vol. 21, no. 4, pp. 3072–3108, 4th Quart., 2019.

13. O. Gurewitz, N. Gradus, E. Biton, and A. Cohen, "Exploring reinforcement learning for scheduling in cellular networks," Mathematics, vol. 12, no. 21, p. 3352, Oct. 2024.

14. 3GPP, "Study on channel model for frequencies from 0.5 to 100 GHz," 3rd Generation Partnership Project, Sophia Antipolis, France, Tech. Rep. TR 38.901, ver. 17.0.0, Jun. 2022.

15. A. Raffin, A. Hill, A. Gleave, A. Kanervisto, M. Ernestus, and N. Dormann, "Stable-baselines3: Reliable reinforcement learning implementations," *J. Mach. Learn. Res.*, vol. 22, no. 268, pp. 1–8, 2021.

Information about the authors:

Yedil Nurakhov – Researcher as Al-Farabi Kazakh National University (Almaty, Kazakhstan. ORCID iD: 0000-0003-0799-7555).

Abzal Kyzyrkanov – Doctor of Philosophy, Professor (Assistant) at Astana IT University (Astana, Kazakhstan. ORCID iD: 0000-0002-8817-8617).

Aldabergen Syrym – JSC "Digital Development Center of the National Bank of Kazakhstan" (Almaty, Kazakhstan. ORCID iD: 0009-0005-6278-8758).

Submission received: 20 May, 2026
Revised: 11 June, 2026
Accepted: 11 June, 2026

IRSTI 28.23.02 https://doi.org/10.26577/jpcsit42202610

Alisher Kaziz[1*], Richard Harrison[2]

[1]Astana IT University, Astana, Kazakhstan
[2]University of Surrey, Guildford, UK
*e-mail: alisher.kaziz@gmail.com

KNOWLEDGE FUSION THROUGH DATALESS BERT PARAMETER MERGING VIA A REGRESSION-BASED AVERAGING

Abstract. Recent progress in dataless knowledge fusion methods has focused on migrating knowledge from a teacher neural network to a student neural network without utilizing the teacher model's training data. As a result, fine-tuning pre-trained language models (PLM) has emerged as a simple method to improve the performance of Natural Language Processing (NLP) models in specific domains. These refined models are available as open source, but typically their training datasets are not, due to issues related to data privacy or intellectual property. This sets up an obstruction to combining knowledge from separate models to produce an enhanced single model. In this paper, we investigate the issue of combining separate models developed on distinct training datasets to create a unified model that performs on out-of-domain data for the student model. We suggest a dataless knowledge fusion technique that integrates models within their parameter space, directed by weights that reduce prediction discrepancies between the combined model and the separate models. Across a wide range of parameter configurations, our assessment indicates that the suggested approach substantially exceeds the performance of traditional baselines like Fisher-weighted averaging or model ensembling. Additionally, we observe that our approach serves as a viable alternative to multi-task learning, capable of maintaining and enhancing the individual models without requiring access to the training data. Our experiments confirm the dataless merging methods, integrating finely adjusted parameters to achieve robust multitask performance with negligible impact on class ratio degradation, approximately 2.1 %.

Keywords: NLP, RegMean, Knowledge Fusion, Dataless NLP, Kazakh NLP, BERT.

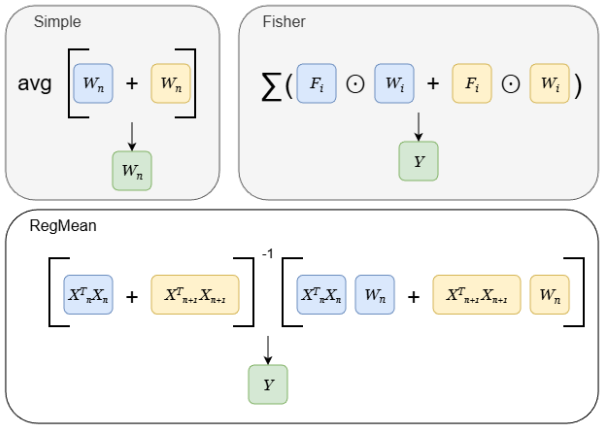
1. Introduction

Current trending paradigm for developing NLP model to solve tasks in specialized domains is to fine-tune a PLM, such as BERT, on task-specific labelled data [1]. This approach outperforms PLM, but it requires producing separate models for each language, domain, and task, which is excessive to maintain and difficult to deploy in real-world pipelines [2], [3]. These challenges tend to arise during the design of NLP models in low-resource settings, such as legal corpora, which are often protected by confidentiality constraints, where annotation is expensive, and cross-lingual transfer is limited by distribution changes of corpus and terminology mismatches [4], [5]. The use of multi-domain and multi-lingual data can partially solve these problems through shared representations across tasks and languages. However, several problems are associated with this approach [6]. First and foremost, centralised exposure to the underpinning labelled datasets is frequently essential for joint training, which may not be possible due to administrative, regulatory, or security constraints [7]. Following that, discovering which task combinations help, could become computationally prohibitive as the number of candidate sources grows [8]. These constraints motivate research in dataless approaches that are able to combine PLMs knowledge that are encoded in independently fine-tuned models without revisiting the original training data.

Model merging offers such a mechanism by combining multiple models directly in parameter space, producing a single model that can be used for inference across domains or languages without additional training [9]. In the context of cross-lingual knowledge fusion, merging is an appealing alternative to ensembling, it retains the development simplicity of a single model while enabling the integration of complementary expertise from multiple sources [10]. Moreover, merging is compatible with privacy-preserving workflows (such as federated or siloed fine-tuning), in which input models exchange parameters or updates rather than the underlying training data [10], [11].

Recent research has clarified both the promise and the limitations of dataless parameter merging for modern NLP systems [12]. Firstly, work on model soups and related weight-averaging approaches show that combining independently fine-tuned checkpoints can yield consistent gains when the models share the same architecture and originate from the same pre-trained initialization, suggesting that parameter-space fusion can work as a cheap alternative to retraining [13], [14]. Secondly, a line of work on task arithmetic and editing vectors indicates that certain task-specific changes can be approximated as linear directions in weight space, enabling controlled composition of capabilities under compatibility assumptions [15]. Thirdly, there are merging methods that incorporate curvature or dependency information [16]. For example, Fisher-weighted merging and regression-based merging such as Regression Mean (RegMean) and its extensions tend to be more robust-compared to naive averaging when source models are heterogeneous (Fig. 1). RegMean shows better account for interference between parameters in dynamics of weight changes to corpus ratio [2], [16].

Figure 1
Comparison between Simple, Fisher, and RegMean parameters merging methods



At the same time, large-scale evaluations on open-weight large language models highlight that merging heuristics are not universally reliable and can fail to transfer across model families, motivating careful validation in each target setting [17]. In parallel, multilingual evaluation efforts continue to emphasise persistent gaps for low-resource languages, even when using strong multilingual pre-trained models [18]. We also combine findings from semantically similar language (Turkish to Kazakh) to estimate the probability of label classes for Weighted-F1 [19], [20]. Together, these findings motivate our choice of a closed-form, interference-aware merging approach (RegMean) and our focus on cross-lingual and domain robustness in the domain [2].

The limitation can be attributed to secrecy, licensing limits, or organisational regulations, and it is typical in legislative materials, which may include delicate private or professional data [21], [22]. Our objective is to fuse NLP models by exploiting knowledge currently contained in present checkpoints, without the need for new labelled information or joint reconfiguration, to provide a single operational model that supports Kazakh legal NLP. Legal language makes cross-linguistic generalisation more difficult. Writing for legal purposes is characterised by extensive and complicated phrases, numerous allusions to legislation and case identification, and varying formulaic structures by jurisdiction [13]. The lack of high-quality annotated legal data sources and tooling in Kazakhstan makes it expensive to develop and upkeep distinct models for each activity and subsystem [18]. A dataless merging strategy combines tuned models into a single model while maintaining a straightforward implementation mechanism [23]. This study is guided by the following research questions (RQs):

- RQ1: How sensitive is the merged model to the choice and number of source checkpoints (e.g., Kazakh-only, multilingual, or legal vs. general-domain)?
- RQ2: What is the impact of merging on in-domain accuracy and out-of-domain robustness?

We only consider merging checkpoints for models that use the base architecture and start with the same pre-trained model. This makes it easy to align the model parameters and merge them directly. This choice keeps the parameters aligned and allows closed-form merging. Studies show engineering teams utilise PLMs [2], [18]. They take a pre-trained BERT model and create different versions for specific tasks or languages. A shared pre-trained BERT checkpoint is taken as a base, and multiple task- or language-specific variants are then produced. Within these conditions, the method suits teams that already run several fine-tuned models and want an inexpensive way to unify them

into a single model for use in various domain processing pipelines.

In this paper, we study Knowledge Fusion via Dataless BERT Parameter Merging with a focus on Kazakh legal modelling. We adopt RegMean as a practical, closed-form weight-merging method that can be applied to multiple fine-tuned checkpoints sharing the same architecture and initialization. We investigate how RegMean can fuse knowledge from models trained on different languages to yield a single BERT-based model that (i) maintains strong in-domain performance, (ii) improves robustness under cross-lingual and domain shift, and (iii) remains parameter-efficient compared to maintaining separate tuned models. Our contributions are three-fold:

1. we formulate a dataless cross-lingual model merging setup for legal NLP with an emphasis on Kazakh language;

2. we provide a systematic empirical evaluation of RegMean-based BERT parameter merging for cross-lingual and domain transfer in legal tasks;

3. we analyse the practical trade-offs of merged models;

2. Materials and methods

This section describes the data sources, model setup, and experimental protocol used to assess dataless parameter merging for Kazakh legal NLP. We summarize the source checkpoints, the RegMean merging procedure, and the downstream evaluation settings so that the experiments can be reproduced and interpreted in a consistent way. Our workflow follows three stages: (i) independently fine-tune multiple BERT checkpoints on language- and domain-specific datasets, (ii) merge the resulting parameters in a dataless manner using RegMean, and (iii) evaluate the merged model on Kazakh legal NLP tasks under both in-domain and cross-domain or cross-lingual conditions (Fig. 2). Importantly, the merging stage does not require any labelled data.

We use a BERT encoder as the shared backbone. All source checkpoints are required to have the same architecture (number of layers, hidden size, attention heads) and the same pre-trained initialization. This constraint ensures that corresponding tensors have identical shapes and can be merged directly in parameter space. We construct a pool of source models by fine-tuning the common backbone on datasets that differ in language (Kazakh, and multilingual) and domain (legal and general domain). Each source model is trained independently within its data silo, only the final model parameters are used for merging. We evaluate on Kazakh legal NLP tasks relevant to downstream document processing. Depending on availability, these include classification of legal documents or sentences, named entity recognition (NER) for legal entities, for which we use KazBERT and supplement it with our own manually labelled legal dataset to better cover domain-specific entities [24]. For each task, we define an in-domain test set and, when possible, an out-of-domain or cross-lingual test set to assess robustness.

Let set $\{\theta_i\}_{i=1}^{M}$ denote the parameters of M fine-tuned source models sharing initialization θ_0, and define $\Delta\theta_i = \theta_i - \theta_0$. RegMean computes a merged update $\Delta\theta$ by minimizing the functional discrepancy between intermediate representations.

$$\Delta\theta^* = \arg\min_{\Delta\theta} \sum_{i=1}^{M} \mathbb{E}_{x\sim\mathcal{D}} \left[\left\| f_{\theta_0+\Delta\theta}(x) - f_{\theta_0+\Delta\theta_i}(x) \right\|_2^2 \right] \quad (1)$$

assume that for a fixed layer the representation is locally linear around θ_0:

$$f_{\theta_0+\Delta\theta}(x) \approx f_{\theta_0}(x) + J(x)\Delta\theta \quad (1)$$

where $J(x)$ is the Jacobian of the layer output with respect to parameters. Then the objective in equation (1) is convex in $\Delta\theta$ and its unique minimiser is:

$$\Delta\theta^* = (\mathbb{E}_{x\sim\mathcal{D}}[J(x)^\top J(x)])^{-1} \left(\frac{1}{M} \sum_{i=1}^{M} \mathbb{E}_{x\sim\mathcal{D}} [J(x)^\top J(x)\Delta\theta_i] \right) \quad (2)$$

that $\mathbb{E}_{x\sim\mathcal{D}}[J(x)^\top J(x)]$ is positive definite in equation (2). Under the linear approximation,

$$f_{\theta_0+\Delta\theta}(x) - f_{\theta_0+\Delta\theta_i}(x) = \\ = J(x)(\Delta\theta - \Delta\theta_i) \tag{3}$$

then, substituting into equation (1) yields

$$\mathcal{L}(\Delta\theta) = \\ \sum_{i=1}^{M} \mathbb{E}_{x\sim\mathcal{D}}\left[(\Delta\theta - \Delta\theta_i)^\top J(x)^\top J(x)(\Delta\theta \right. \\ \left. - \Delta\theta_i)\right] \tag{4}$$

and were we define equation (4) as

$$H = \mathbb{E}_{x\sim\mathcal{D}}[J(x)^\top J(x)] \tag{5}$$

then,

$$\mathcal{L}(\Delta\theta) = \sum_{i=1}^{M} (\Delta\theta - \Delta\theta_i)^\top H(\Delta\theta - \Delta\theta_i) \tag{6}$$

Since H is positive semidefinite, the objective is convex. If H is positive definite, the objective is strictly convex and admits a unique minimiser.

Taking the gradient and setting it to zero:

$$\nabla_{\Delta\theta}\mathcal{L} = 2\sum_{i=1}^{M} H\,(\Delta\theta - \Delta\theta_i) = 0 \tag{7}$$

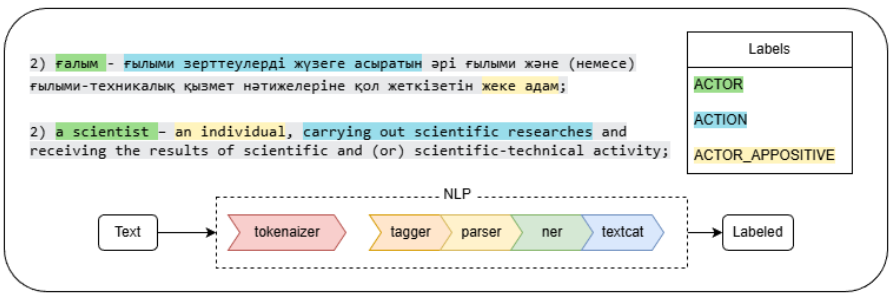
this implies,

$$MH\Delta\theta = H\sum_{i=1}^{M} \Delta\,\theta_i \tag{8}$$

Multiplying both sides by H^{-1} yields the stated solution. In practice, expectations over $\mathcal{D}$ are approximated using a representative unlabelled corpus drawn from the target domain. Merging is performed once per source set and does not involve additional gradient-based training.

We compare RegMean to common alternatives: (i) selecting the best single source checkpoint per task, (ii) uniform weight averaging over source checkpoints (model soups), and (iii) prediction ensembling when applicable. These baselines separate gains from parameter-space fusion versus multi-model inference. Reported fine-tuning methods (optimizer, learning rate schedule, batch size, number of epochs, maximum sequence length) and model selection criterion ensured comparability, and a consistent fine-tuning procedure was implemented across source models within each experimental setting. We use standard task metrics (e.g., accuracy and macro-F1 for classification; entity-level weighted-F1 for NER). In addition to average performance, we analyse cross-lingual transfer to Kazakh language, robustness under domain shift. We apply a consistent NLP pipeline. This includes document segmentation, sentence splitting, and normalisation of common legal artifacts (e.g., article references, case numbers, dates, and monetary amounts). Where the downstream task benefits from it, we optionally anonymize personally identifiable information (PII) with placeholders to reduce leakage and to better simulate privacy-preserving deployment constraints. Since Kazakh language can be sensitive to subword segmentation quality, we specify the tokenizer used for all experiments and keep it fixed across source checkpoints and merged models. When source checkpoints use different tokenizers, we treat them as incompatible and do not merge them (Fig.2).

Figure 2
Tokenizer for all evaluated models in cross lingual domain to keep minor bias on NER and entity relation tasks



Study defines criteria for selecting source checkpoints for merging, including minimum validation performance, domain relevance, and language coverage. To mitigate negative transfer, we exclude checkpoints that are substantially misaligned with the target domain and report sensitivity to these selection decisions. All experiments are conducted with fixed random seeds. To analyse the conditions under which merging succeeds or fails, we examine (i) sensitivity to the number and composition of source checkpoints, (ii) robustness to variations in fine-tuning hyper-parameters, and (iii) qualitative error patterns on Kazakh legal data.

3. Results

Macro-F1 characterises class-balanced effectiveness by weighting each class equally. Weighted-F1 include observed class frequencies and therefore reflects performance under label imbalance. While we introduce MCC, which provides a correlation-based measure that remains informative when legal label distributions are skewed, validation across the metrics reported in Table 1. RegMean-merged model shows the strongest aggregate performance. When Macro-F1 assessment was conducted to measure specialised model performance with tuned RegMean model, the comparative analysis shows modest improvement in various scenarios. For instance, average improvements by 2.9 points with MCC by 0.05 was indicated. Even if prediction performance was enhanced, study reveals test scenarios where classes was not balanced specialised model outperform RegMean approach. Similar behaviour was revealed in Jin's research [2]. One reason why unbalanced classes scenario have declined, because it is RegMean method nature that merges parameters based on casual regression. The most likely causes of F1 poor gains is underestimation of parameters weights in RegMean merging process. The fact that Weighted-F1 is going up means that RegMean is still doing well with the majority class and it is also getting better at handling the minority labels.

In the Table 1 each model results are reported as the mean standard deviation across multiple random seeds. Higher Macro-F1 and MCC values correspond to better class-balanced effectiveness and greater overall consistency of predictions.

Table 1
Performance comparison of specialised models and tuned models with RegMean approach for the Kazakh legal document classification task

Model	Macro F1 (± std)	Weighted F1 (± std)	MCC (± std)
KazBERT	83.9 ± 0.6	85.2 ± 0.5	0.81 ± 0.01
Multilingual Legal BERT	81.5 ± 0.7	83.1 ± 0.6	0.78 ± 0.02
Uniform Weight Averaging	84.4 ± 0.5	85.9 ± 0.8	0.83 ± 0.01
Prediction Ensemble	85.7 ± 0.1	87.0 ± 0.3	0.85 ± 0.01
RegMean (Merged Model)	**86.8 ± 0.3**	**87.3 ± 0.2**	**0.86 ± 0.01**

The essence of RegMean approach is to combine models parameters to cover under-tuned space. Models fused together performed notable results in various test scenarios. While common methods of models tuning require holding each version and running many checks to breach the bottleneck of predictions thresholds, on other than RegMean method uses just one set of settings and gets results that are modest or outperform common methods in dataless environment. This shows that combining settings from models can bring together useful knowledge from models trained on different languages and areas without needing the original training data. RegMean approach is designed to perform in a specific area and keeps steady balanced across different domain categories by focusing on inter-domain accuracy and maintaining class-balanced results. Merged model keeps results steady on parent tuned models datasets and attains confident results on new task in a edge of the parent models domains. RegMean comes close to the effectiveness of prediction ensembling, while it retains the operational convenience of deploying a single model. Obtained results indicate that merging increases in-domain accuracy while preserving class-balanced performance across domain categories, with RegMean approach. As reported in Table 1, MCC shows an 8% relative gain over the Legal BERT baseline, alongside an approximately 5% relative improvement in Macro-F1. Earlier work on Fisher-weighted merging has typically

documented relative F1 increases of about 2 to 3% in heterogeneous scenarios, whereas the present cross-lingual legal setting exhibits a moderately larger effect size [16]. On the other hand, while gathered metrics reveal improvements model performance tuned with RegMean approach further imply that the benefits are not confined to majority classes. Instead, they point to broader gains in learned representations and in the quality of the induced decision boundaries. The relative improvement in minority-class F1 (RI_c)with respect to the baseline is summarised in Table 2:

Table 2
Per-class F1 comparison between KazBERT and RegMean on the legal classification task

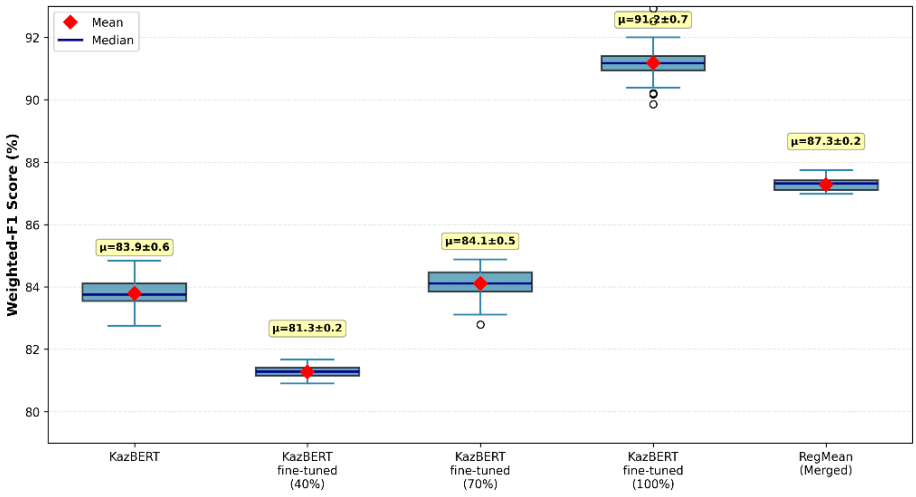
Class	Sample size	KazBERT F1	RegMean F1	RI_c %
PERSON	13577	88.2 ± 0.1	88.7 ± 0.2	+0.6
LAW	5693	84.2 ± 0.2	89.1 ± 0.3	+5.9
NORP	3714	88.7 ± 0.5	87.3 ± 0.3	-1.0
PROJ	2111	84.7 ± 0.2	86.4 ± 0.1	+2.1

where,

$$RI_c = \frac{F1_{RegMean} - F1_{Baseline}}{F1_{Baseline}} \times 100\% \qquad (9)$$

In the Table 2, relative Δ reports the percentage change of RegMean over KazBERT for each label, highlighting where merging improves minor classes. Study shows minor improvements in couple classes, performance increasing caused from one of source model which specialised on law labelling. Our study also reports a comparison of Weighted-F1 scores for Kazakh legal classification using box plots as shown in Figure 3. We look at how the KazBERT model works when it is given amounts of labelled data to learn from. We compare the KazBERT model with versions that have been fine-tuned using 40%, 70% and 100% of the available training data and also the RegMean merged model. The box plots show how the results are spread out when we use random seeds, which helps us see both how well the model works on average and how much the results can vary. When we have labelled the data, the KazBERT model learns from it better. If Kazakh legal classification KazBERT model is trained using 40% of the dataset Kazakh legal classification KazBERT model gets a Weighted-F1 score of 81.3 give or take 0.2. In Figure 3 median line indicates typical performance, while boxes and whiskers show variability across random seeds.

Figure 3
Box plots of Weighted-F1 under different data regimes (40%, 70%, 100%), the baseline KazBERT model, and the RegMean merged model

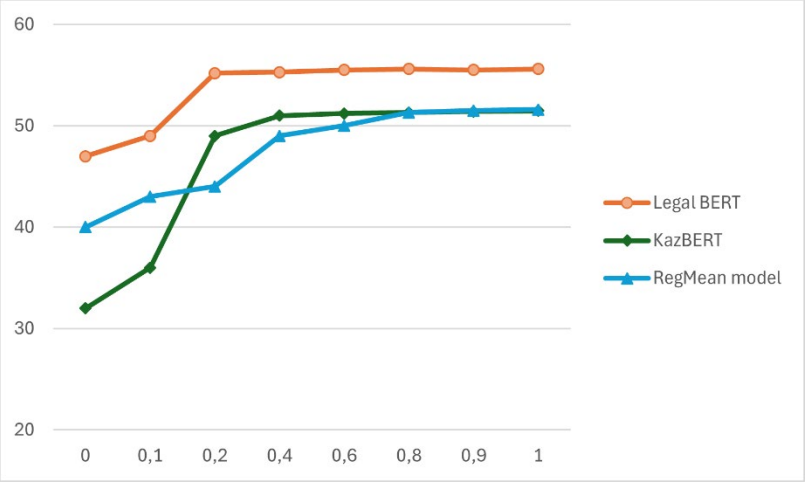


Box plot was introduced to show comparison of training data shortage and dataless RegMean approach. The box plot examine a range of values and short lines suggesting that the results are similar throughout the runs. However, this similarity means that the model is not very good at generalizing when it does not have data to learn from. The model does not learn well because it has not seen different types of legal issues and ways of writing. When we add data to 70% the models performance gets much better reaching 84.1 with a small variation of 0.5. The box plot for this data shows that the middle value is higher and the range of values is wider compared to when we had 40% of the data. However, even if variance across seeds does not mean the model explores more solutions, this wider range applicable in term of possible decision variance affected by low-resources settings. Even though the model does better with data it also becomes more sensitive to how it is set up and random changes.

Tuned KazBERT model on 100% of dataset outperforms RegMean tuned approach. Using all the data gives the results with a performance of 91.2 and a variation of 0.7. This supports our first research question. The box plot shows that the middle value of Weighted-F1 is higher than with data. However the range of values is also the widest among the models that were fine-tuned. The wider range of values and longer lines in the box plot show that the models performance varies across different runs. This is what happens when engineer tune a model, then they suggest that more data helps it to performance increasing on a task but it also makes the model more sensitive to changes [17]. The model could be tuned to bypass validation tests, however its performance varies more, between runs on various datasets.

KazBERT model shows modest scoring 83.9 with some room for improvement, common method to improve performance is to adjust dataset, this modest results was shown in between the 70% and 40% configurations in terms of dataset volume from complete one. The results for this model vary a bit. When we merge the model using RegMean, it scores 87.3. This model is stable with results close to the average, which indicate close or low variance among probes. It works well as fine-tuning 40% of the model but it shows underperformance of accuracy. RegMean does not outperform the model that's fully fine-tuned but it is a good balance, between accuracy and stability. Compared to a model that's 70% fine-tuned RegMean is more accurate and stable. RegMean perform notable improvements in term of performance and it has less variation by compared to the KazBERT model. RegMean is 3.4 points better and has a lower standard deviation. The 100% fine-tuned model achieves the highest peak performance but also shows the greatest variability (±0.7). In tested environments, such variability can translate into unpredictability during retraining or model updates. In contrast, RegMean demonstrates consistent performance across seeds (±0.2), indicating robustness to stochastic training effects.

Figure 4
NER performance as a function of the RegMean mixing coefficient α, the curve highlights the stability region where merging yields consistent gains



The narrow box distribution suggests that parameter-space merging aggregates complementary information from multiple checkpoints in a manner that smooths individual optimisation noise. This variance reduction is consistent with findings in large language models merging studies, where merged checkpoints demonstrate lower run-to-run dispersion compared to independently fine-tuned models [17]. In our case, standard deviation decreases from ± 0.7 in full-data fine-tuning to ± 0.2 in the merged model. This behaviour can be interpreted as a form of implicit regularization. To evaluate NEW performance we applied mixing coefficient approach for both our baseline models and RegMean tuned model, this approach fit to quantify sensitivity in sequence labelling (Fig. 4). Steady point and a plateau after α 0.7 indicate that the merged model is not overly sensitive to the exact parameters of baseline models but benefit from weighting elements. This suggests that RegMean provides a stable operating region in which practitioners can select α without extensive hyper-parameter tuning, while still retaining gains over individual baselines. Fine-tuning adapts parameters along a single optimisation trajectory, potentially converging to different local minima depending on initialisation. Parameter merging, by contrast, combines independently learned updates, effectively averaging task-relevant representations while attenuating idiosyncratic deviations. The resulting model occupies a more central region of the loss landscape, which may explain its reduced variance.

4. Discussion

The analysis of data efficiency shows a striking difference between RegMean and standard tuning on the reduced datasets. In particular, the training KazBERT on 40% of the entire data results in about 6 point drop in performance compared to the RegMean merged model. Even increasing the training set to 70% does not help, as the tuned model is still three points behind. This is somewhat ironic because the cross-lingual transfer in low-resource already usually yields around 2 to 5 points improvement.

Dataless parameter merging aside, this clearly demonstrates that the merge of parameters without data is also a way to get labelled data that is not available and it is the most effective way to take advantage of the knowledge from different models. While fine-tuning on a complete dataset ultimately yields the best performance, compiling and annotating extensive legal datasets is cost-prohibitive, especially for low-resource languages such as Kazakh. Therefore, unlike the common methods, RegMean provides solution that is both competitive and repeatable across runs.

The box-plot analysis reveals the following important points:

- First, the more training data we use, the better performance we can attain, but the greater the variability in the performance.
- Second, RegMean is effective in the trade-off between accuracy and stability. It surpasses the performance of fine-tuning with less data while also having significantly less variance.
- Third, the combination of the parameters makes the final training process less sensitive to the initial optimization conditions.

The implications of this observation are that not merely is dataless parameter merge an approach for acquiring knowledge of parameters, it is a method for increasing the robustness of a model. When data is extremely scarce in the labelled set, the justified power of consistent low-variance results is significant.

The NER α-sweep (Fig. 4) is yet another indicator of robustness in results. We see a fairly broad performance plateau, which implies that the advantages of merging are not overly hyper-sensitive coefficient tuning. This is a significant advantage in deployment when applying merged models to new tasks or domains. This is naturally consistent with our results in classification tasks, confirming that sequence labelling tasks benefit by a similar level of robustness.

5. Conclusions

The consolidation of cross-lingual knowledge in low-resource legal NLP environments where training data is extremely limited is a key topic of this study. We designed a dataless merging framework that directly integrates independently trained BERT checkpoints in the parameter space through RegMean strategy. Notably, this approach does not require shared datasets or joint retraining which is often computationally costly.

Our experiments on Kazakh legal classification have demonstrated that the interference aware merging approach consistently outperforms individual specialist models and baseline parameter

averaging methods. By generating one single checkpoint for deployment, this method not only shows the advantages of model ensembles but also retains tunability. The resulting merged model displayed a significant rise in both Macro-F1 and MCC, as well as a decreased variance in different random seeds.

We indicate that the variance decline is because parameter-space fusion acts as a kind of implicit regularisation, which suppresses optimization noise and stops checkpoints trained on different data from interfering destructively. Our findings not only focus on the increase in raw accuracy but also highlight the fact that dataless merging serves as a stabilizer for limited supervision. On the contrary, it often surpasses traditional fine-tuning on reduced data sets and as such significantly closes the gap in performance with models trained on complete data. These features are essential for applications in low-resource languages or in fields that suffer from high costs due to data annotation. Moreover, the method presents a valid option for situations that face restrictions from either strict privacy regulations or organizational regulations thus excluding discreet sharing of training data.

This research confirms that even though dataless parameter merging cannot be regarded as a complementarity technique, it still is an excellent method for the transfer of knowledge between different models without any reduction in functionality or efficiency. The use of interference aware merging strategies has been shown to be a very efficient alternative to retraining on a central source, especially in cases when the access to data is limited or privacy is a matter of concern. The incorporation in legal domain NLP with RegMean parameter fusion in the early stages of the development is most likely to provide a multi-domain NLP tasks tuning with benefits from it. The statement is particularly relevant for low-resource domains that face the resource scarcity and data absence challenges, such as having under-labelled datasets, small validation sample sizes, or being completely dataless.

Acknowledgments

We would like to express our sincere gratitude to Dr Bolatzhan Kumalakov for his valuable guidance and support throughout this research.

Funding

This research received no external funding.

Conflicts of Interest

The authors declare no conflict of interest.

Author Contributions

Conceptualization, A.K.and R.H.; methodology, A.K. and R.H.; software, A.K.; validation, A.K.; formal analysis, A.K.; resources, A.K. and R.H.; writing-original draft preparation, A.K.; writing-review and editing, A.K. and R.H.; visualization, A.K.; supervision, R.H.

References

1. D. Khurana, A. Koli, K. Khatter, and S. Singh, “Natural language processing: state of the art, current trends and challenges,” *Multimedia Tools and Applications*, vol. 82, no. 3, pp. 3713–3744, Jul. 2022, doi: 10.1007/s11042-022-13428-4.

2. X. Jin, X. Ren, D. Preotiuc-Pietro, and P. Cheng, “Dataless knowledge fusion by merging weights of language models,” in *Proceedings of the 11th International Conference on Learning Representations*, pp. 1 – 19, Dec. 2023, doi: 10.48550/arxiv.2212.09849

3. H. Zheng, L. Shen, A. Tang, Y. Luo, H. Hu, B. Du, Y. Wen, and D. Tao, “Learning from models beyond fine-tuning,” *Nature Machine Intelligence*, vol. 7, no. 1, pp. 6–17, Jan. 2025, doi: 10.1038/s42256-024-00961-0.

4. X. Jiang, W. Wang, S. Tian, H. Wang, T. Lookman, and Y. Su, “Applications of natural language processing and large language models in materials discovery,” *Npj Computational Materials*, vol. 11, no. 1, Mar. 2025, doi: 10.1038/s41524-025-01554-0.

5. S. Joshi, M. S. Khan, A. Dafe, K. Singh, V. Zope, and T. Jhamtani, “Fine Tuning LLMs for Low Resource Languages,” in *5th International Conference on Image Processing and Capsule Networks (ICIPCN)*, pp. 511–519, Jul. 2024, doi: 10.1109/icipcn63822.2024.00090.

6. K. Liu, N. Uplavikar, W. Jiang, and Y. Fu, “Privacy-Preserving Multi-task Learning,” in *IEEE International Conference on Data Mining (ICDM)*, Nov. 2018, doi: 10.1109/icdm.2018.00147.

7. M. Tao, C. Zhang, Q. Huang, T. Ma, S. Huang, D. Zhao, and Y. Feng, “Unlocking the Potential of Model Merging for Low-Resource Languages,” in *Findings of the Association for Computational Linguistics*, Nov. 2024, pp. 8705–8720, doi: 10.18653/v1/2024.findings-emnlp.508.

8. Y. Hu, Y. Yao, N. Zhang, H. Chen, and S. Deng, “Exploring model kinship for merging large language models,” in *Findings of the Association for Computational Linguistics*, Nov. 2025, doi: 10.48550/arxiv.2410.12613.

9. C. Poth, J. Pfeiffer, A. Rücklé, and I. Gurevych, “What to Pre-Train on? Efficient intermediate task selection,” in *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 10585–10605, Jan. 2021, doi: 10.18653/v1/2021.emnlp-main.827.

10. B. Y. Lin, C. He, Z. Ze, H. Wang, Y. Hua, C. Dupuy, R. Gupta, M. Soltanolkotabi, X. Ren, and S. Avestimehr, “FEDNLP: Benchmarking Federated Learning Methods for natural language processing tasks,” in *Findings of the Association for Computational Linguistics*, pp. 157–175, July 2022, doi: 10.18653/v1/2022.findings-naacl.13.

11. J. Zhang, S. Vahidian, M. Kuo, C. Li, R. Zhang, T. Yu, G. Wang, and Y. Chen, “Towards Building The Federatedgpt: Federated Instruction Tuning,” in *IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 6915–6919, Mar. 2024, doi: 10.1109/icassp48485.2024.10447454.

12. J. Wang, M. Kulkarni, and D. Preotiuc-Pietro, “Multi-Domain Named Entity Recognition with Genre-Aware and Agnostic Inference,” in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 8476–8488, Jan. 2020, doi: 10.18653/v1/2020.acl-main.750.

13. Y. Wang, Y. Gu, Y. Zhang, Q. Zhou, Z. Yan, C. Xie, X. Wang, J. Yuan, and H. Yang, “Model merging scaling laws in large language models,” *arXiv (Cornell University)*, Sep. 2025, doi: 10.48550/arxiv.2509.24244.

14. G. Ortiz-Jimenez, A. Favero, and P. Frossard, “Task Arithmetic in the tangent Space: Improved editing of Pre-Trained models,” in *Proceedings of the 37th International Conference on Neural Information Processing System,* May 2023, doi: 10.48550/arxiv.2305.12827.

15. G. Ilharco, M. T. Ribeiro, M. Wortsman, S. Gururangan, L. Schmidt, H. Hajishirzi, and A. Farhadi, “Editing Models with Task Arithmetic,” in *International Conference on Learning Representations*, May. 2023, doi: 10.48550/arxiv.2212.04089.

16. M. Matena and C. Raffel, “Merging Models with Fisher-Weighted Averaging,,” in *Proceedings of the 36th International Conference on Neural Information Processing Systems*, pp. 17703–17716, Nov. 2022, doi: 10.48550/arXiv.2111.09832.

17. E. Yang, L. Shen, G. Guo, X. Wang, X. Cao, J. Zhang, and D. Tao, “Model Merging in LLMs, MLLMs, and beyond: methods, theories, applications, and opportunities,” *ACM Computing Surveys*, vol. 58, no. 8, pp. 1–41, Jan. 2026, doi: 10.1145/3787849.

18. N. Kadyrbek, Z. Tuimebayev, M. Mansurova, and V. Viegas, “The Development of Small-Scale Language Models for Low-Resource Languages, with a Focus on Kazakh and Direct Preference Optimization,” *Big Data and Cognitive Computing*, vol. 9, no. 5, p. 137, May 2025, doi: 10.3390/bdcc9050137.

19. C. Çetindağ, B. Yazıcıoğlu, and A. Koç, “Named-entity recognition in Turkish legal texts,” *Natural Language Engineering*, vol. 29, no. 3, pp. 615–642, Jul. 2022, doi: 10.1017/s1351324922000304.

20. M. Zampieri, P. Nakov, and Y. Scherrer, “Natural language processing for similar languages, varieties, and dialects: A survey,” *Natural Language Engineering*, vol. 26, no. 6, pp. 595–612, Nov. 2020, doi: 10.1017/s135132492000049.

21. F. Ariai, J. Mackenzie, and G. Demartini, “Natural Language Processing for the legal domain: A survey of tasks, datasets, models, and challenges,” *ACM Computing Surveys*, vol. 58, no. 6, pp. 1–37, Nov. 2025, doi: 10.1145/3777009.

22. P. Tulajiang, Y. Sun, Y. Zhang, Y. Le, K. Xiao, and H. Lin, “A Bilingual Legal NER Dataset and Semantics-Aware Cross-Lingual label transfer Method for Low-Resource languages,” *ACM Transactions on Asian and Low-Resource Language Information Processing*, vol. 24, no. 9, pp. 1–21, Jul. 2025, doi: 10.1145/3748325.

23. D. Shome and K. Yadav, “EXnet: Efficient In-context Learning for Data-less Text classification,” *arXiv (Cornell University)*, May 2023, doi: 10.48550/arxiv.2305.14622.

24. R. Yeshpanov, Y. Khassanov, and H. A. Varol, “KazNERD: Kazakh Named Entity Recognition Dataset,” in Proceedings of the Thirteenth Language Resources and Evaluation Conference, pp. 417 – 426, June 2022, doi: 10.48550/arXiv.2111.13419.

Information about the authors:

Alisher Kaziz – PhD student (EP Computer Science) at the School of Software Engineering, Astana IT University. His research interests include multiagent systems, explainable artificial intelligence, and machine learning knowledge. He focuses on the development of interpretable AI models for machine learning knowledge fusion to improve decision making robustness (Astana, Kazakhstan, e-mail: alisher.kaziz@gmail.com. ORCID iD: 0009-0003-2554-6149).

Dr Richard Harrison – Director of Learning and Teaching for the Faculty of Engineering and Physical Sciences Foundation Year Programmes at the University of Surrey. He also Chairs the Faculty's Mathematics Education Working Group and is a Fellow of the Institute of Mathematics and its Applications. He has extensive experience as a Mathematician and Computer Scientist in industry and academia both in the UK and overseas (e-mail: r.harrison@surrey.ac.uk. ORCID iD: 0009-0005-5206-1117).

Submission received: 11 May, 2026
Revised: 15 June, 2026
Accepted: 15 June, 2026

CONTENTS