Received: xxx **Accepted:** xxx **Published:** xxx.
DOI. xxxxxxx
Volume xxx, Issue xxx, p.p. 1-50: xxx
Research Article

Open Access

Control and Optimization in Applied Mathematics - COAM

E-SNM-CNN-LSTM: Enhanced Star-Nosed Mole Optimization for Traffic Forecasting and Energy-Efficient Base-Station Sleep Control

Hamid Rahimi, **Reza Sheibani** ✉, **Gelareh Veisi**

Department of Computer Engineering, Ma.C., Islamic Azad University, Mashhad, Iran

✉ **Correspondence:**
Reza Sheibani
E-mail:
reza.sheibani@iau.ac.ir

Abstract. In this paper, a hybrid framework called E-SNM-CNN-LSTM was proposed for cellular traffic prediction and base station sleep-wake decision support in heterogeneous networks. First, traffic data were processed using an entropy-based refinement step to reduce noise and short-term spikes. Then, a combination of CNN and LSTM with the adaptive memory gate (AMG) was employed to extract spatial-temporal patterns. Meta-parameter tuning was subsequently performed using a newly proposed Star-nosed mole (SNM) algorithm. The performance of the proposed method was evaluated on four datasets, Milan, Madrid LTE, Telecom Shanghai, and Trentino, and compared with baseline models and the Transformer–ConvLSTM reference framework. Results showed that the proposed model provided small and stable improvements in MAE, RMSE, and R^2 for traffic prediction, while achieving higher accuracy and a lower false-off rate in sleep decision-making. In high-density scenarios such as Telecom Shanghai and sparse-data conditions such as Trentino, the framework improved both prediction accuracy and energy-based decision-making quality. Furthermore, simulations of a two-layer macro-small cell network showed that the proposed model increased energy savings and reduced the number of on/off switches. Independent evaluation of the SNM algorithm on the CEC BC 2017 benchmark functions also demonstrated competitive and stable performance in search capability and overall ranking. Overall, the findings indicate that the proposed framework is a practical and reliable solution for efficient energy management in 5G and beyond networks, supporting sustainable operation and intelligent resource utilization across heterogeneous wireless communication environments while maintaining robust performance under diverse network traffic conditions.

How to Cite
Rahimi, H., Sheibani, R., Veisi, G. (2027). "E-SNM-CNN-LSTM: Enhanced star-nosed mole optimization for traffic forecasting and energy-efficient base-station sleep control". *Control and Optimization in Applied Mathematics*, X(x), 1–50. https://doi.org/10.30473/coam.2026.78962.1446

Keywords. Traffic prediction, Cellular networks, Evolutionary algorithms, SNM algorithm, Neural network.

MSC. 90C59; 68T07; 90B18.

©2026 by the authors. Lisensee PNU, Tehran, Iran. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution 4.0 International (CC BY4.0) (http:/creativecommons.org/licenses/by/4.0)

1 Introduction

In heterogeneous 5G networks and beyond, base station energy consumption has become a major factor in operational cost and network stability. Among the various strategies to reduce this consumption, dynamic and intelligent switching of base stations based on traffic conditions has a special place [26, 37, 47, 61]. However, the success of this strategy is entirely dependent on the accuracy of network load forecasting, because any error in traffic estimation can lead to unnecessary outages, increased latency, degraded service quality, and instability between active and sleep states [22, 48, 61]. In the 6G network landscape, the growth of scale, diversity of services, and the need for near-real-time decision-making have increased the importance of traffic forecasting more than ever before; thus, this prediction has become a fundamental component for energy management, resource allocation, and quality of service assurance [15, 21, 58, 63]. Accordingly, research has moved from simple threshold-based methods to data-driven models and deep learning. Review studies show that models such as LSTM, spatiotemporal mixed networks, and graph-based approaches provide higher accuracy than classical methods in many scenarios [48, 68]. However, these models are usually associated with challenges such as high computational complexity, sensitivity to sudden traffic changes, difficulty in generalizing in dynamic environments, and limited implementation at the edge [47, 68].

Among these approaches, hybrid architectures such as CNN+LSTM have attracted much attention due to their ability to simultaneously extract local features and temporal patterns. These structures have provided a good balance between prediction accuracy and computational cost in many network and communication applications [9, 66]. However, previous research experience shows that the final performance of these models is highly dependent on the tuning of parameters and meta-parameters. In practice, even if the appropriate architecture is chosen, suboptimal parameter tuning can lead to loss of accuracy and instability in energy-driven decisions [5, 43]. Therefore, some studies have used evolutionary and metaheuristic algorithms to tune the metaparameters of deep models and reported significant improvements in error metrics [30, 65].

However, in the field of cellular traffic forecasting, and especially in frameworks designed to support base station sleep strategies, a coherent framework that includes evolutionary parameter tuning as an independent and purposeful part of the CNN+LSTM architecture is still lacking [22, 56]. Compared with the closest related study, by Vengaimarbhan and Rajinigirinath [56], the proposed framework introduces two main novelties. First, the prediction backbone is redesigned from a ConvLSTM-based structure to a lighter CNN–LSTM architecture, which reduces computational overhead while preserving the ability to model spatiotemporal traffic patterns. Second, the key hyperparameters and adaptive-memory coefficients are not tuned manually or empirically; instead, they are optimized by the proposed Star-Nosed Mole (SNM) algorithm. Therefore, the main novelty of this paper lies in combining entropy-based prepro-

In Press

cessing, CNN–LSTM feature extraction, AMG-based memory adaptation, and SNM-driven hyperparameter tuning within a unified framework for traffic forecasting and base-station sleep control. The closest related work in the literature is the study by Jiang et al. [23], but the Transformer–ConvLSTM baseline used for comparison is the model of Vengaimarbhan and Rajinigirinath [56]. In this work, the model of Vengaimarbhan and Rajinigirinath [56] is treated strictly as the baseline model, while the study by Jiang et al. [23] is cited as related work without implying reuse of its methods. Accordingly, this paper presents a sleep prediction and decision-making framework for base stations, in which a CNN+LSTM submodel is used to simultaneously extract spatial and temporal traffic features, and its meta-parameters are automatically tuned by the proposed evolutionary algorithm "Star-Nosed Mole Algorithm". The aim of this approach is to maintain prediction accuracy while enabling practical deployment in real 5G/6G networks.

The main research gap addressed in this study is the lack of a lightweight and deployable framework for cellular traffic prediction that jointly considers prediction accuracy, computational cost, and energy-aware sleep control. To address this gap, the present work contributes: (i) entropy-based traffic refinement, (ii) a CNN–LSTM backbone with adaptive memory gating, (iii) SNM-based hyperparameter tuning, and (iv) a unified evaluation of forecasting accuracy, sleep-control reliability, and energy saving.

In Press

2 Literature Review

2.1 Traffic Prediction as a Foundation for Energy Management

A review of the research conducted shows that reducing energy consumption in cellular networks, especially through dynamic and intelligent sleep of base stations, has increasingly become dependent on accurate traffic prediction [37, 38, 48, 61]. In dense networks, power-on and power-off decisions will only be efficient if they are based on a correct understanding of the network load pattern; otherwise, inappropriate power-offs and frequent oscillations between active and sleep modes will both reduce the quality of service and weaken energy efficiency [37, 61].

In new network generations, especially with the move towards 6G, this issue has become more important, because the high scale of the network, the heterogeneity of services, and the need for rapid response have made energy management a problem dependent on prediction [21]. For this reason, traffic forecasting is no longer just an analytical tool, but is part of the infrastructure of network decision-making.

2.2 Transition from Classical Methods to Deep Learning

In the past, many traffic forecasting methods relied on fixed rules, predetermined thresholds, and simple statistical models. Although these approaches were acceptable in simple environments, they lacked sufficient accuracy and stability in the dynamic and changing scenarios of today's networks [48]. This has led to the focus of research gradually shifting towards data-driven models and deep learning.

Deep models have a high ability to extract complex patterns from large data sets and can better learn nonlinear relationships between features and output [5, 22]. In the networking domain, these models have also been used for tasks such as traffic forecasting, channel estimation, resource allocation, and network behavior analysis [21, 30, 46, 66]. However, despite the increased accuracy, these models usually require higher computational costs and are not always robust to sudden traffic spikes or noisy data [5, 16].

2.3 Sequence-based Models and the Role of LSTM

Since network traffic inherently has temporal patterns, LSTM has become one of the most common choices for predicting this type of data [48]. In some works, LSTM has been used to predict dynamic demand in networks beyond 5G and its output has been used in network management decisions [27]. Also, in edge-oriented environments, LSTM-based encoder–decoder models have been used for multivariate traffic prediction [28]. Deep learning approaches have also been applied to related temporal problems in mobile networks, such as predicting traffic peaks [32] and personalizing predictions across sites through layer-wise federated learning [29].

However, LSTM alone has limitations when dealing with very complex patterns or long sequences [43]. For this reason, many researchers have moved towards combining it with other architectures to both improve accuracy and overcome some of the limitations of purely temporal models [5, 22].

2.4 Combining CNN and LSTM for Spatiotemporal Data

In addition to the temporal dimension, network traffic data also has spatial and local structures. For this reason, combining CNN and LSTM has been proposed as one of the effective solutions for analyzing this data [22, 57]. In this architecture, CNN first extracts local and structural features, and then LSTM models temporal patterns using this compact representation.

This approach has also had positive results in communication applications. For example, in some studies related to channel estimation and telecommunication data analysis, the combi-

In Press

nation of CNN and LSTM has shown good accuracy and stability [57]. Also, reviews related to the application of CNN in communication show that this model performs well in extracting features from complex data [30, 46]; more broadly, CNN-based architectures have shown strong feature-extraction capability in other data-intensive domains such as image segmentation, which supports their adoption for spatiotemporal traffic modeling [40]. In the specific context of cellular traffic, CNN–LSTM combinations enhanced with frequency-domain filtering [18] and lightweight multiservice, multimodal feature fusion [31] have both been reported to improve prediction quality while controlling model size. Therefore, the use of CNN+LSTM for traffic prediction is conceptually and empirically quite justified, especially in scenarios where both spatial and temporal correlation are important.

2.5 Graph-based and Attention-based Approaches

In recent years, graph-based models have been considered to more accurately capture the dependencies between cells and network points [3]. In these models, the network is represented as a graph and the relationships between nodes are learned with the help of graph neural networks. This approach can capture spatial–temporal relationships better than many classical models, but in return is usually associated with higher computational cost and complexity [16]. Several recent studies illustrate this trend in cellular and general traffic settings, including state-transition graph attention networks for cell-level prediction [51], spatial-temporal Chebyshev graph networks for IoT-based intelligent transportation systems [64], attention-based spatial-temporal graph networks [45], and dual-branch spatial-temporal graph neural networks with attention for cellular traffic [8].

Similarly, attention-based and transformative models have been used to improve traffic forecasting [41, 58]. These models perform well in capturing long-range dependencies, but due to their training complexity and high computational requirements, their implementation in edge environments or in resource-constrained systems is not always straightforward [58]. Lightweight hybrid-attention networks [53] and diffusion-convolutional GRU models combined with multi-head attention [62] have both been proposed to mitigate this cost while retaining most of the accuracy benefit. Hence, although these architectures are attractive in terms of accuracy, they still pose practical deployment and implementation cost issues.

2.6 Using Traffic Prediction in Base Station Sleep Strategies

With the strengthening of the role of traffic prediction, a significant part of the research has directly applied this prediction to base station sleep decisions. In some studies, CNN–LSTM

In Press

hybrid architectures have been used for load prediction and base station sleep control, and energy consumption reduction has been reported [22]. Data-driven sleeping strategies built directly on traffic prediction have also been proposed and shown to reduce energy use in both homogeneous and heterogeneous cellular deployments [33, 59, 70], and stochastic models of the sleep/wake process itself have been used to analyze energy efficiency in 5G base stations [24]. Graph-based traffic prediction has additionally been coupled with reinforcement learning for joint load balancing and sleep control [36]. In some other works, reinforcement learning or self-organizing methods have also been proposed for base station sleep control, but these methods usually either require a lot of simplification or are accompanied by limited guarantees in optimality [30, 46].

In more recent studies, hybrid approaches based on transformers and ConvLSTM have also been proposed for base station sleep control, which provide better accuracy, but the computational complexity and the need for fine-tuning of parameters are still their main challenges [49]. Methods based on multi-attribute prediction, dynamic switching, and variable thresholds have also been proposed, but on a large scale, their sensitivity to settings and management overhead are significant [35, 44].

Zhang et al. [68] proposed an inverted-transformer-based framework for mobile traffic forecasting and dual-level base-station sleep control. They demonstrate the strength of transformer-based models in capturing long-range dependencies and supporting sleep decisions. However, transformer-based approaches are typically more computationally demanding and less suitable for lightweight deployment. In contrast, the present study uses a CNN–LSTM backbone with adaptive memory gating and SNM-based tuning to achieve a more practical balance between accuracy, complexity, and real-time deployability.

2.7 The Role of Parameter Tuning and Evolutionary Algorithms

One of the important findings of the literature is that the performance quality of deep learning models is seriously dependent on the tuning of meta-parameters [19, 60]. In many cases, even if the model structure is appropriate, the inappropriate choice of the number of layers, filter sizes, number of hidden units, learning rate, or input window size can cause a loss of accuracy.

To deal with this problem, various studies have used evolutionary and meta-heuristic algorithms to tune models. These methods have led to improved accuracy and stability in various fields, including wind speed prediction, air pollution prediction, tool life, intrusion detection, and industrial production prediction [19]. A similar pattern has been reported outside the networking domain as well; for example, a CNN–LSTM model tuned by an improved sparrow search algorithm has been used for oil-well production prediction, with the metaheuristic search again yielding more stable and accurate results than manual tuning [69]. Also, in network traf-

In Press

fic prediction, the use of genetic optimization to tune graph models has reported positive results [19].

However, most of these studies have either focused on graph-based models or used meta-parameter tuning as a side effect; as a result, a coherent framework that uses evolutionary tuning as a core part of the CNN+LSTM architecture in cellular traffic prediction and base station sleep decision-making has received less attention [56].

Recent studies confirm the usefulness of evolutionary optimization for tuning deep models. For example, Ampratwum and Nayak [3] applied genetic-algorithm-based hyperparameter optimization in cellular traffic prediction, and Andi et al. [4] used chaotic particle swarm optimization for LSTM tuning. These studies support the view that metaheuristic search can enhance robustness and accuracy of deep prediction models.

2.8 Summary of the Literature Review and Research Direction

The literature review shows that three main conclusions can be drawn in this area. First, traffic prediction is the basis of energy-based decision-making in next-generation networks, and without it, dynamic and intelligent base station sleep cannot lead to sustainable savings [48].

Second, deep learning models, especially CNN, LSTM, and graph-based or attention-based methods, have improved the prediction accuracy; however, computational cost, complexity, and difficulty of practical deployment remain major obstacles [41].

Third, the tuning of meta-parameters plays a decisive role in the final performance of the model, and the use of evolutionary algorithms can make the search in the parameter space more efficient and the results more stable [19, 34].

Accordingly, the main research gap is that there is still no unified, lightweight, and deployable framework for cellular traffic prediction that both takes advantage of the CNN+LSTM architecture and tunes its parameters optimally. For this reason, the present paper, relying on a CNN+LSTM submodel and the proposed evolutionary algorithm “Star-Nosed Mole”, and considering the model presented in [56], seeks to provide a method that can both increase prediction accuracy and be suitable for use in base station sleep strategies in real 5G and 6G networks.

Although Ref. [56] is the closest related work, our approach remains lighter and more deployable thanks to the CNN–LSTM backbone and SNM-based hyperparameter tuning. The proposed entropy refinement and adaptive memory gating (AMG) further balance accuracy and efficiency, enabling joint evaluation of traffic forecasting and energy-aware sleep decisions under multiple urban scenarios.

In Press

3 Proposed E-SNM-CNN-LSTM Model

In this study, a cellular traffic prediction and base station sleep control framework for heterogeneous 5G networks and beyond is presented, which is based on the combination of a lightweight spatiotemporal model and an evolutionary optimization mechanism. The main idea of the proposed method is that, instead of relying on heavy and expensive models, a CNN+LSTM submodel is used to simultaneously learn spatial and temporal patterns, and along with it, the metaparameters are automatically tuned by the proposed evolutionary algorithm of the star-nosed mole. This design is inspired by dynamic traffic and sleep prediction frameworks in cellular networks, but tries to balance prediction accuracy, computational complexity, and edge deployment [22, 58].

The overall architecture of the system consists of several main parts: data preprocessing, entropy-based traffic filtering, spatial feature extraction with CNN, temporal dependency modeling with LSTM, and finally the decision layer for base station sleep. In this structure, first the raw traffic data is prepared to reduce the strong noise and meaningless fluctuations as much as possible. Then, the local features and structures embedded in the data are extracted by the convolutional network and its output is transferred to the LSTM to learn short-term and medium-term temporal patterns. This approach takes advantage of the capacity of CNN in extracting compact representations and the power of LSTM in modeling sequences [54, 66].

The proposed framework, titled E-SNM-CNN-LSTM, is designed to predict cellular traffic and support the base station sleep strategy. This framework is inspired by spatiotemporal and attention-based hybrid models, but has two key differences: first, it uses CNN+LSTM as the core of prediction, instead of just using heavier structures such as ConvLSTM [56]; second, it does not leave the tuning of meta-parameters to manual or empirical use, but rather leaves it to an intelligent evolutionary process.

In this research, a framework named “E-SNM-CNN-LSTM” is proposed for cellular traffic prediction and dynamic sleep strategy implementation for base stations. The main idea of this framework is that:

- First, the raw traffic is refined with an entropy-based coding step to attenuate purely random fluctuations and enhance meaningful patterns;
- Then, spatial patterns are modeled by a convolutional network and the time-varying trend is modeled by an LSTM;
- A transformer-inspired attention mechanism highlights important time steps;
- Finally, the architectural parameters and coefficients of the memory gates are automatically adjusted by the evolutionary algorithm of the star-nosed mole (SNM).

In Press

In the following, the new algorithm of the star-nosed mole is first presented, and then the steps of the proposed model are discussed.

3.1 Star-Nosed Mole Algorithm (SNM)

The star-nosed mole is found in eastern Canada and the northeastern United States. Its habitat is lowland and damp areas and it feeds on aquatic insects, worms, mollusks, and small fish. The animal's skin is covered with black waterproof fur and it has a large tail and legs. It is between 15 and 20 centimeters long, weighs 55 grams, and has 44 teeth. The 22 fleshy tentacles at the end of the animal's snout distinguish it from other mole rats. It is believed that these highly sensitive tentacles identify food by detecting electric currents while hunting. The star-nosed mole requires only 230 milliseconds from detection to consumption of prey, making it the fastest mammal.

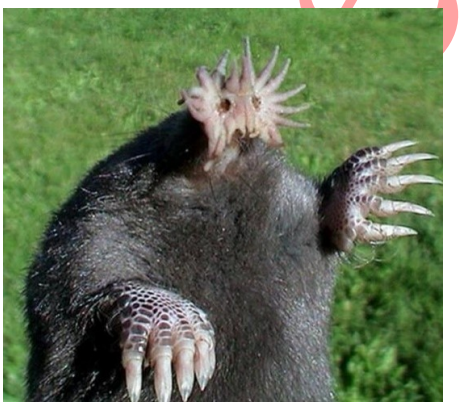

Figure 1: Star-nosed mole

This number may seem a bit excessive, but with 100,000 nerve fibers in each fiber of the star-nosed mole's brain, it can process all the data. This mammal's tactile sensor is five times greater than that of humans, allowing it to hunt, walk, and swallow at high speed. The star-nosed mole can recognize and swallow prey within 120 milliseconds. For this reason, it holds the Guinness World Record [17] for the fastest swallow among mammals, a finding also documented in the scientific literature [11, 14].

The star-nosed mole is a small, semi-aquatic vole found in the lowlands of northern North America [10]. It has over 25,000 sensory receptors in its tactile organs, known as Eimer's organs, and is about the size of a hamster. With the help of its Eimer's organs, it senses seismic waves [67].

The star-nosed mole lives in lowlands and wet areas, feeding on small invertebrates such as aquatic insects and terrestrial insects [7], and has also been found in dry grasslands further

from water and on the other side of the mountains as high as 1676 m. However, the star-nosed mole prefers wet areas and swamps [52].

Not much is known about the reproductive cycle of the star-nosed mole. They are more social than other moles and live in small colonies, and it is believed that pairs stay together to mate throughout the winter. After mating, there are usually four to five offspring in late spring or early summer. The breeding season begins in early spring, and the female produces only one litter per year, unless the first litter is unsuccessful [71].

3.1.1 Summary of the Star-Nosed Mole's Behavior in the Stages of the Proposed Algorithm

The proposed algorithm is inspired by the biological and behavioral characteristics of this animal. This algorithm takes advantage of the sexual differentiation of males and females, and the reproduction and birth of 4 to 5 offspring at a time, to diversify the population, and applies mechanisms of elitism and replacement of the best solutions.

This animal lives in both terrestrial and aquatic environments. In the terrestrial environment, it uses its tentacles to search for food in the soil. To do this, 10 pairs of tentacles search different areas and the 11th pair of tentacles makes the final detection. In the aquatic environment, this animal finds food by creating bubbles around the search area.

In the algorithm, 10 neighbors are created around each search area (a possible solution) and if a neighbor has a better fitness (using the 11th pair of tentacles to calculate fitness) than the current area, that neighbor is selected as the next area to search. The way the soil is searched and the neighbors are created is accompanied by changes (parameter w) such that initially smaller areas are searched and over time the search areas to find food become larger (exploratory behavior). In search of food in water, this animal creates bubbles around the search site and selects the next search site by smelling these bubbles. The algorithm also considers the generation of bubbles (10 bubbles) with random and decreasing behavior (parameter O), which initially searches a larger area and over time this area becomes smaller (exploitative behavior). This animal pays attention to its caloric intake in choosing food, and this behavior is achieved in the algorithm with a pair of 11 tentacles that are used to calculate fitness and move towards the best fitness (foraging theory).

Steps of the Star-Nosed Mole Algorithm.

Step 1. Create the initial population of the star-nosed mole with Equation (1), which is represented by the matrix equivalent in Equation (2).

In Press

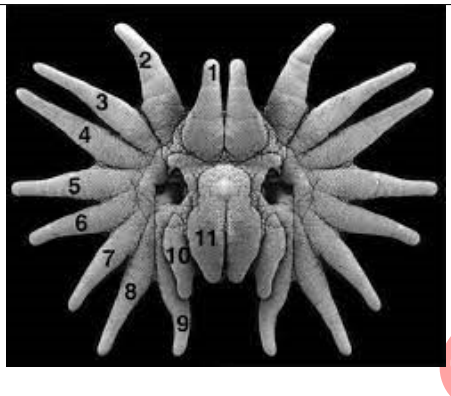

Figure 2: Pair of tentacles of a star-nosed mole

Specify the initial population size by generating females as between 45% and 55% of the population using a random function and generating the remaining number of the population as males. In Equation (1), the floor function specifies an integer between 0.45 and 0.5 of the total population N as females (N_f) and the rest as males (N_m):

$$N_f = \lfloor 0.5 - 0.05 \times \text{rand} \times N \rfloor, \qquad N_m = N - N_f, \tag{1}$$

where N_f is the number of female mice, N_m is the number of male mice, N is the total number of mice, and rand is a random function that returns a number between 0 and 1.

The population is represented by the position matrix

$$X = \begin{bmatrix} x_1^1 & \cdots & x_d^1 \\ \vdots & \vdots & \vdots \\ x_1^N & \cdots & x_d^N \end{bmatrix}. \tag{2}$$

Step 2. Evaluate the objective function (using the 11th pair of tentacles for detection) and identify the best answer as the bait (the highest fitness means the bait with the highest caloric value, which must be eaten, and the mouse moves to that area).

Bait position. For each agent (mouse) we have 10 positions; 2 positions with higher fitness are selected and considered as bait. The update is performed assuming both are baits, and then the average between the two answers is taken; the answer obtained becomes the new position. The objective function matrix is

$$\text{fitness} = \begin{bmatrix} \text{fitness}_1 \\ \vdots \\ \text{fitness}_N \end{bmatrix}. \tag{3}$$

Step 3. The star-nosed mole searches for food in both soil and water environments. It is assumed that the probability of searching for prey in soil and water is 50%. Therefore, the selection of neighbors (search points) is performed in one of two cases for each mouse:

In Press

$$\text{Search\_}X(j) = \begin{cases} \text{Search\_}X(j)_1, & P \leq 0.5, \\ \text{Search\_}X(j)_2, & P > 0.5, \end{cases} \tag{4}$$

where Search_$X(j)_1$ is related to soil search and Search_$X(j)_2$ is related to water search.

Search in soil (exploratory behavior). Creating 10 neighbors (a fixed number of 10 neighbors, or search points, corresponding to 10 tentacles) is done by moving the tentacles of the mouse's snout with parameter Δ_1:

$$\text{Search\_}X(j)_1 = X(i) + \Delta_1, \tag{5}$$

where Search_$X(j)_1$ represents the location of the jth search point in the soil (searching with each tentacle in the snout). Small values of Δ_1 result in a small-range search, while large values allow for a global search:

$$\Delta_1 = \left(\text{rand} \cdot X_{\text{rand}} - \text{rand} \cdot x(i)\right) \cdot \left\| \frac{\text{Bait\_}x_1 + \text{Bait\_}x_2}{2} - x(i) \right\| \cdot w. \tag{6}$$

In Equation (6), the two search points that have the highest fitness and their average are used to calculate Δ_1 (the tentacles of the mouse's snout always move towards the best food sources with higher caloric value, and this is taken into account by using the average of the two search points with the highest fitness).

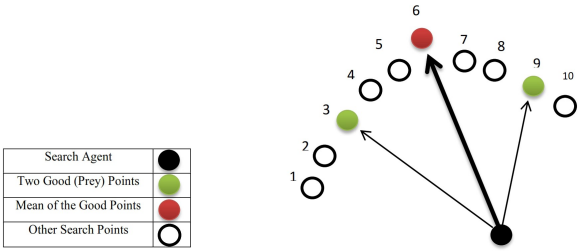

Figure 3: Land search (survey of 10 surrounding points, using the average of two potential prey positions)

As the algorithm iterations progress and the search agents Bait_x_1 and Bait_x_2 converge, they become closer to each other (the distance between the best solutions found decreases because the best solutions are around the original optimum of the problem), or similar. Therefore, at the end of the algorithm, a move towards a bait (search point) occurs, thus helping the algorithm converge to the optimum of the problem. (Thus, using the average does not harm the convergence of the algorithm; for example, if a search agent is at position 4 and another is at position 2, the average is position 3, and the algorithm moves towards position 3.)

Here, "rand" is a random number with a uniform distribution between 0 and 1, X_{rand} is the random position of a search agent, w is a nonlinear weight controlling the search range of the snout. In this regard, the first term shows that $X(i)$ moves to a random position (X_{rand}). For

In Press

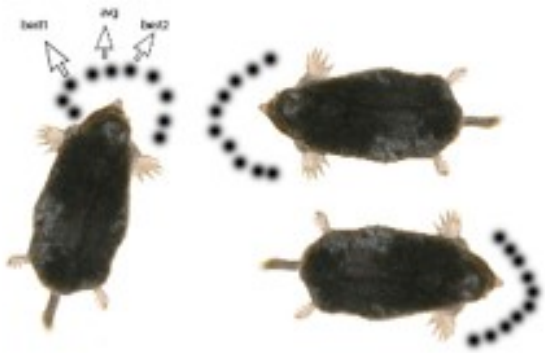

Figure 4: Soil search

the second term, with increasing iteration, $X(i)$ approaches the mean $(\text{Bait\_}x_1 + \text{Bait\_}x_2)/2$ and the Euclidean distance between the positions of the baits with higher caloric value and $X(i)$ (the current position of the search point) decreases.

In Equation (6), the presence of the random coefficient rand and the use of the random position of the search point X_{rand} ensure diverse behavior of the algorithm when searching regions of the problem space. w in the third term is given by

$$w = \text{randn} \cdot \left(\text{iter} \frac{\text{rand}}{\text{Max}_{\text{iter}}} \right), \tag{7}$$

where randn is a random vector with a normal distribution with mean 0.5 and standard deviation 1, rand is a random number with a uniform distribution between 0 and 1, iter is the current iteration number, and Max_{iter} is the final iteration number. In Equation (7), the value of w increases with the number of rounds of the algorithm, so that the selection of neighbors (search points) starts from the nearest points and moves toward the farthest points.

Water search (exploitative behavior). The star-nosed mole sniffs out prey by randomly creating bubbles around the prey area. The creation of search points is a neighborhood search in water with the effect of Δ_2 in Equation (9). The creation of 10 neighbors (a fixed number of 10 neighbors, or search points, corresponding to 10 bubbles) is done by smelling bubbles with parameter Δ_2. (This animal produces between 8 and 12 bubbles per second and searches the underwater space to find prey.)

$$\text{Search\_}X(j)_2 = X(i) + \Delta_2, \tag{8}$$

$$\Delta_2 = \|\text{Bait\_}x_1 - x(i)\| \cdot O + \text{Bait\_}x_1, \tag{9}$$

$$O = \text{randn} \cdot e^{b\left(\text{iter} \frac{\text{rand}}{\text{Max}_{\text{iter}}} \right)} \cdot \sin\left(2\pi \left(\text{iter} \frac{\text{rand}}{\text{Max}_{\text{iter}}} \right) \right). \tag{10}$$

In Press

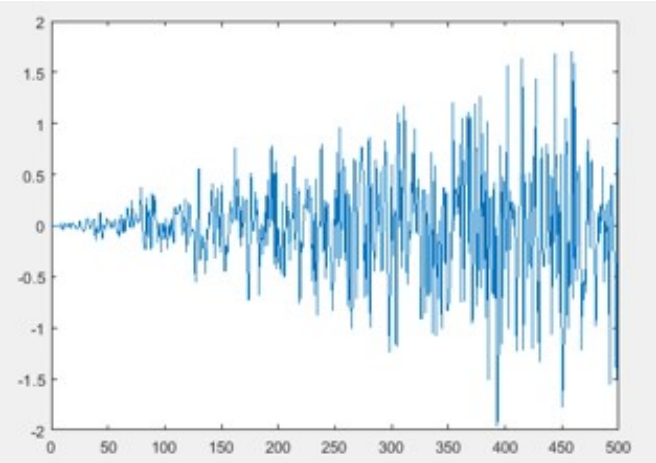

Figure 5: How w changes over iterations

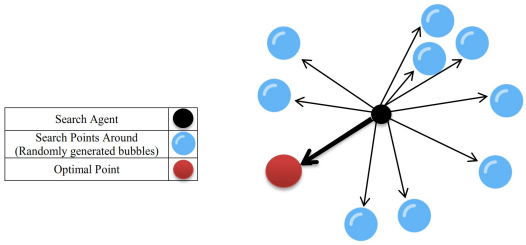

Figure 6: Water search (360 degrees)

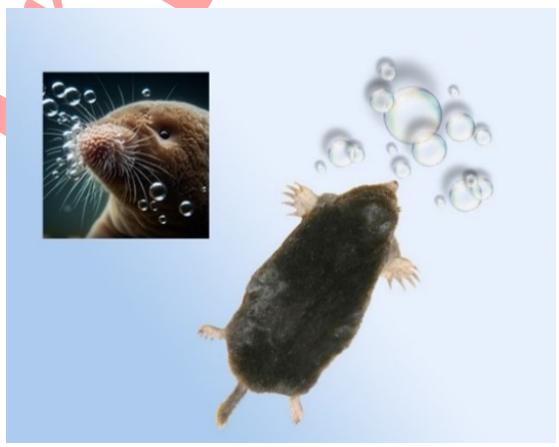

Figure 7: Water search

In Equations (8)–(10), Δ_2 is the distance of the mouse's current point to the best point (bait), Bait_x_1 is the best answer obtained (bait), and $x(i)$ is the current search point. Search_$X(j)_2$ represents the position of the jth search point in the water (search with bubbles). The parameter O is a random behavior with the effect of a sine term and a decreasing exponential function, and causes the search to move from large areas to smaller areas over time. The value of b is a random variable in the range $[-1, 1]$. Searching in water is an exploitative behavior because the distance of the current solution to the best solution is the criterion for movement, and in this direction the effects of the O parameter are decreasing.

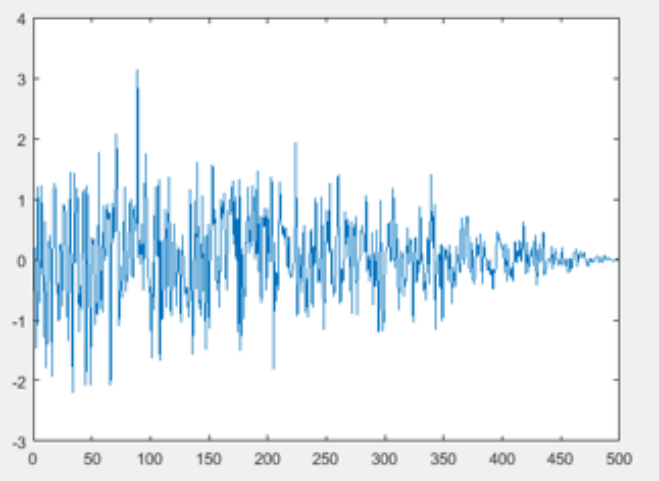

Figure 8: How O changes over iterations

Step 4. Determine the objective function value for each search point (search_point) in the neighborhood and determine the best search point.

Step 5. If the best search point in the neighborhood has a better objective function value than the current search point, execute Step 6; otherwise, proceed to Step 7.

Step 6. Update the search velocity vector based on Equations (11)–(12) and create a new search point position based on Equation (13). The search velocity vector is

$$V = \begin{bmatrix} v_1 & \cdots & v_d \end{bmatrix}, \tag{11}$$

$$V = \text{randn} \cdot \frac{\text{fitness}(i) - \text{search\_point\_fitness}(j)}{\|x(i,d) - \text{search\_point\_}x(j,d)\|}, \tag{12}$$

where the random number "randn" generates different solutions and increases the global search. The ith search point vector is considered relative to the jth search point in the neighborhood, fitness(i) and search_point_fitness(j) represent the objective value of search point i and neighboring search point j, respectively, and d represents the problem dimension. Equation (13) determines the new position of the search point:

$$\text{new}X(i) = X(i) + V_i, \tag{13}$$

where new$X(i)$ represents the new position of search point i. The vector of a search agent in the new state includes the vector of that agent in the previous state plus the velocity value.

Updating the position. The star-nosed mole algorithm randomly selects another search point $LF(x)$ (a Lévy-flight function representing the exploration direction). If the objective function of this search point is more optimal than that of the current search point $X(i)$, the search continues in the same direction of movement; otherwise, it changes direction toward the best prey found so far:

$$\text{if} \quad \text{fitness}(LF) < \text{fitness}(i): \quad \text{new}X(i) = X(i) + \text{randn} \cdot \big(LF(x) - X(i)\big), \tag{14}$$

$$\text{else}: \quad \text{new}X(i) = X(i) + \text{randn} \cdot \big(\text{Bait\_}X - X(i)\big),$$

$$LF(x) = 0.01 \times \frac{u \times \sigma}{|v|^{1/\beta}}, \tag{15}$$

$$\sigma = \left(\frac{\Gamma(1+\beta) \times \sin\left(\frac{\pi\beta}{2}\right)}{\Gamma\left(\frac{1+\beta}{2}\right) \times \beta \times 2^{(\beta-1)/2}} \right)^{1/\beta}, \qquad \Gamma(x) = (x-1)!.$$

Reproduction. Performing the mating operator, which can occur between each male and a female: in this move, each male mates with a female within the reproductive radius R (Equation (16)) and four offspring are produced. The reproductive radius has a decreasing trend and decreases with the number of iterations of the algorithm; at the beginning of the algorithm, the production of offspring is higher, and at the end of the algorithm it decreases (in the life of this animal, reproduction occurs more when there is more food and less when food resources are reduced, because with reduced food resources, mice must move away from other mice to find food, which in turn reduces reproduction). The selection of the female is performed by examining the fitness of the remaining mice, such that the fitness is calculated for each female and the female mouse with the highest fitness is more likely to be selected for reproduction.

The reproduction of this animal is between 4 and 5 offspring, and in the algorithm 4 offspring are produced from 2 parents. The production of 4 offspring increases the population of search agents; of the 4 offspring produced, those that are better than their parents (have better fitness) replace them, and the two weaker offspring are eliminated (eliminating solutions with low fitness, i.e., elitism). For each female mouse, the fitness is calculated and a roulette-wheel selection is used according to fitness:

$$R = \frac{U_b - L_b}{\left(\text{Max}_{\text{iter}} - \text{iter}\right) + \varepsilon}, \tag{16}$$

$$\text{offspring}_{j,1} = \beta_1 \times \text{Male}_j + (1 - \beta_1) \times \text{Female}_j^i, \tag{17}$$

In Press

$$\text{offspring}_{j,2} = (1-\beta_1) \times \text{Male}_j + \beta_1 \times \text{Female}_j^i, \quad (18)$$

$$\text{offspring}_{j,3} = \beta_2 \times \text{Male}_j + (1-\beta_2) \times \text{Female}_j^i, \quad (19)$$

$$\text{offspring}_{j,4} = (1-\beta_2) \times \text{Male}_j + \beta_2 \times \text{Female}_j^i. \quad (20)$$

Here, $\text{offspring}_{j,1}$ through $\text{offspring}_{j,4}$ are the first through fourth offspring of the mating, and β_1 and β_2 are random numbers with a uniform distribution between 0 and 1. In Equations (16)–(20), U_b is the upper bound of the variables, L_b is the lower bound of the variables, $\varepsilon = 0.0001$, and β_1 and β_2 are random numbers with a normal distribution with mean 0.5 and standard deviation 1.

Step 9. Substitute offspring for parents if they are better, and eliminate weak solutions.

Step 10. Compute the objective function for the search agents.

Step 11. Check the termination condition. If the number of iterations is exhausted, return the optimal solution; otherwise, repeat Steps 3 to 10.

Pseudocode of the Star-Nosed Mole Algorithm. The SNM algorithm can be interpreted mathematically as a population-based search process that alternates between exploration and exploitation (See Figure 9). The soil-search operator promotes global exploration by generating candidate neighbors around the current solution, while the water-search operator intensifies local search around promising solutions. The reproduction operator preserves strong candidates and generates new ones through recombination. Boundary handling keeps all candidate solutions feasible, and the termination rule returns the best solution after the evaluation budget is exhausted. Compared with PSO, GWO, and DE, SNM differs in how it balances neighbor-based search, adaptive step control, and reproduction-based diversification.

3.2 Steps of the Proposed Model

The general steps of the proposed model, following the structure in [56], include five modules: preprocessing and entropy coding, spatial feature extraction with CNN, temporal modeling with LSTM and an adaptive memory gate, a transformer attention layer, an SNM optimization layer, and a base station sleep decision layer mounted on top of this entire structure.

3.2.1 Entropy-based Traffic Coding

Mobile network traffic usually behaves erratically and, in many cases, is accompanied by sudden jumps, short-lived fluctuations, and significant noise. If this data is fed to the input of the model without filtering, the model may become overly sensitive to transient changes or fail to

In Press

Begin
Initialize the algorithm parameters
P = [0.1]
Initialize the search agents N_f and N_m by
$N_f = [0.5 - 0.05 \times rand \times N]$
$N_m = N - N_f$
While $iter < Max_{iter}$
For i=1 to N (all agent)
Calculate the fitness of the search agent (minimize)
$Bait\_x1$= 1th best global
$Bait\_x2$= 2th best global
rp random numbers [0.1]
IF rp < P
THEN(exploration)
For j=1 to 10
$W = (randn * \left(iter \frac{rand}{Max_{iter}}\right))$
$\Delta_1 = \left(rand * Xrand - rand * x(i)\right) * \left\|\left(\frac{Bait_{x1} + Bait_{x2}}{2}\right) - x(i)\right\| * w$
$Search\_X(j)_1 = X(i) + \Delta_1$
end for
ELSE (explotain)
For j=1 to 10
$O = randn * (e^{b*(iter\frac{rand}{Max_{iter}})} * \sin(2\pi(iter\frac{rand}{Max_{iter}})))$
$\Delta_2 = \|Bait\_x1 - x(i)\| * O + Bait\_x1$
$Search\_X(j)_2 = X(i) + \Delta_2$
end for
end IF
$search\_point = Search\_X(j)$
Calculate the fitness of neighbors and move towards the best neighbor
IF $search\_point\_fitness(j) < fitness(i)$
THEN
$V = randn * \frac{fitness(i) - search\_point\_fitness(j)}{\|x(i,d) - search\_point\_x(j,d)\|}$
$newX(i) = X(i) + V_i$
ELSE
$\Gamma(x) = (x-1)!$
$\sigma = \left(\frac{\Gamma(1+\beta) \times sin\left(\frac{\pi\beta}{2}\right)}{\Gamma\left(\frac{1+\beta}{2}\right) \times \beta \times 2^{\left(\frac{\beta-1}{2}\right)}}\right)^{\frac{1}{\beta}}$
$LF(x) = 0.01 \times \frac{u \times \sigma}{|v|^{\frac{1}{\beta}}}$
if $_{fitness(LF)} < _{fitness(i)}$
$newX(i) = X(i) + randn * (LF(x) - X(i))$
ELSE
$newX(i) = X(i) + randn * (Bait\_X1 - X(i))$
end IF
end for
(explotain local search) Calculate:
R = $\frac{(Ub-LB)}{(Max_{iter}-iter)+\varepsilon}$
for N_f in neighborhood radius(R) N_m $offspring$ $generation$
$offspring_j\ 1 = \beta_1 \times Male_j + (1-\beta_1) \times Female_j^i$
$offspring_j\ 2 = (1-\beta_1) \times Male_j + \beta_1 \times Female_j^i$
$offspring_j\ 3 = \beta_2 \times Male_j + (1-\beta_2) \times Female_j^i$
$offspring_j\ 4 = (1-\beta_2) \times Male_j + \beta_2 \times Female_j^i$
Eliminate bad search engines and elitism
end for
End while
End

Figure 9: Pseudocode of the star-nosed mole algorithm

recognize real trends. To mitigate this problem, the proposed method uses an entropy-based filtering step.

The goal of this step is to attenuate parts of the data that are less informative or are simply due to random fluctuations, while meaningful and recurring patterns become more prominent. In this way, the final model input is of higher quality and the CNN+LSTM network can learn real traffic patterns more stably. This step is especially important in dense and dynamic cellular environments, because in such conditions short-term fluctuations can severely affect prediction performance.

Suppose that, for a cell and a service (e.g., Internet), the traffic sequence is

$$X = \{x_1, x_2, \ldots, x_n\}, \tag{21}$$

where x_t is the traffic volume in time interval t. Using the empirical distribution of values, the Shannon entropy is calculated as

$$H(X) = -\sum_{t=1}^{n} p(x_t) \log p(x_t). \tag{22}$$

If

$$H(X) \leq \frac{1}{n} \sum_{t=1}^{n} H(X), \tag{23}$$

the traffic in that interval is considered highly unstable and entropy-dependent smoothing is applied:

$$x'_t = \alpha\, x_t + (1 - \alpha)\, x_{t-1}, \tag{24}$$

where α is a coefficient that decreases with increasing entropy (high entropy $\Rightarrow$ stronger smoothing). For sections with low entropy, the value of α is chosen close to 1 to preserve the original waveform as much as possible.

After smoothing, an entropy weight is assigned to each sequence:

$$w(X) = \frac{H(X)}{H_{\max}}, \tag{25}$$

and the weighted sequence is defined as

$$X_{\text{encd}} = w(X) \cdot X. \tag{26}$$

Furthermore, the data are normalized by min–max scaling:

$$X_{\text{norm}} = \frac{x_t - x_{\min}}{x_{\max} - x_{\min}}, \tag{27}$$

and transformed into input–output sequences using a sliding window. If the time window length is L, the training samples are constructed as follows:

In Press

$$
\begin{aligned}
[x_1, \ldots, x_L] &\to y_1 = x_{L+2}, \\
[x_2, \ldots, x_{L+1}] &\to y_2 = x_{L+2}, \\
&\vdots \\
[x_k, \ldots, x_{k+L-1}] &\to y_k = x_{k+L}.
\end{aligned} \tag{28}
$$

For B cells and C service types, the final input data is a three-dimensional tensor $X \in \mathbb{R}^{T \times B \times C}$.

3.2.2 Spatial Pattern Extraction with CNN

After data filtering, convolutional layers are responsible for extracting spatial and local patterns. In the traffic prediction problem, these features can include correlations between neighboring cells, spatial variations in network load, or local structures embedded in the input data. CNN transforms the initial data representation into a set of compact and meaningful features by applying learned filters.

Using CNN in this section has two important advantages. First, it reduces the number of learnable parameters compared to fully connected models and, as a result, reduces the training cost. Second, due to its ability to extract local features, it provides a suitable input for the temporal part of the model. For this reason, CNN acts as a feature extractor in the proposed framework and prepares the necessary spatial information for the next step [30, 46, 57].

In order to understand the relationship between neighboring cells (cells in a city, region, or spatial grid), a convolutional network is first applied on the spatial dimension. At each time point t, a slice of data is considered as

$$
X_t \in \mathbb{R}^{B \times C}. \tag{29}
$$

By applying one-dimensional convolutional filters along the cell dimension, spatial features are extracted:

$$
Z_t = \sigma\big(W_{\text{conv}} * X_t + b_{\text{conv}}\big), \tag{30}
$$

where $*$ is the convolution operator, $\sigma(\cdot)$ is a nonlinear activation function (e.g., ReLU), W_{conv} is the filter matrix, and b_{conv} is the bias. The feature map Z_t represents the combination of the cell's traffic with its nearest neighbors.

By stacking these maps along the time dimension, the spatial feature sequence is obtained:

$$
Z = \{Z_1, \ldots, Z_T\}, \tag{31}
$$

which becomes the input to the LSTM section.

In Press

3.2.3 Temporal Modeling with LSTM and Adaptive Memory Gate (AMG)

The output of the CNN layers is fed into the LSTM network to model temporal dependencies in the traffic data. Since network traffic is inherently a sequential phenomenon, the ability of the LSTM to retain past information and learn the relationship between different time steps plays an important role in the quality of the prediction [5, 27, 48].

In this section, the LSTM attempts to track traffic changes over short and medium time frames and estimate future trends based on past patterns. This feature is particularly useful for base station sleep decisions, as making the right decision in this area depends on knowing the future state of the network load. However, for the model not to be vulnerable to long sequences or sharp changes, its parameters must be carefully tuned, a problem that is solved in this research through an evolutionary algorithm [5, 43].

To track traffic changes over time, LSTM layers are used on the CNN output. If z_t is the flattened spatial feature vector at time t, the standard LSTM equations are

$$f_t = \sigma\big(W_f z_t + U_f h_{t-1} + b_f\big), \tag{32}$$
$$i_t = \sigma\big(W_i z_t + U_i h_{t-1} + b_i\big), \tag{33}$$
$$\widetilde{c}_t = \tanh\big(W_c z_t + U_c h_{t-1} + b_c\big), \tag{34}$$
$$c_t = f_t \odot c_{t-1} + i_t \odot \widetilde{c}_t, \tag{35}$$
$$o_t = \sigma\big(W_o z_t + U_o h_{t-1} + b_o\big), \tag{36}$$
$$h_t = o_t \odot \tanh(c_t), \tag{37}$$

where h_t is the output at time t and c_t is the internal cell state.

In order for this memory to adapt the amount of past information it retains depending on how quiet or busy the traffic is, an "adaptive memory gate" (AMG) is placed on the LSTM gates. The general form of the modified gates is

$$f_t^{\text{AMG}} = \sigma\big(W_f^* z_t + U_f^* h_{t-1} + b_f + \beta_f \Delta_f\big), \tag{38}$$
$$i_t^{\text{AMG}} = \sigma\big(W_i^* z_t + U_i^* h_{t-1} + b_i + \beta_i \Delta_i\big), \tag{39}$$
$$o_t^{\text{AMG}} = \sigma\big(W_o^* z_t + U_o^* h_{t-1} + b_o + \beta_o \Delta_o\big), \tag{40}$$

where $\beta_f, \beta_i, \beta_o$ are the AMG control coefficients; $\Delta_f, \Delta_i, \Delta_o$ are terms calculated from traffic-fluctuation indices (e.g., short-term variance or small-window entropy); and these coefficients β are adjusted by the SNM algorithm.

Thus, the internal state and the new output are updated as

$$c_t = f_t^{\text{AMG}} \odot c_{t-1} + i_t^{\text{AMG}} \odot \widetilde{c}_t, \tag{41}$$
$$h_t = o_t^{\text{AMG}} \odot \tanh(c_t). \tag{42}$$

In Press

During periods when traffic is very volatile, AMG increases the effective forgetting value so that momentary fluctuations do not accumulate in the memory. During stable periods, the gates are adjusted to retain a longer memory of the past.

3.2.4 Transformer-based Attention Layer (TEA)

Although LSTM models the time sequence to a large extent, over long timescales it may not be able to accurately determine which past moments are most important for predicting the future. To enhance this aspect, the output sequence of the LSTM layer is fed to a transformer attention block.

Suppose the last LSTM layer produces the sequence

$$H = [h_1, \ldots, h_T] \in \mathbb{R}^{T \times d_h}. \tag{43}$$

First, three mappings that are common in transformers are applied:

$$Q = W_Q H, \tag{44}$$

$$K = W_K H, \tag{45}$$

$$V = W_V H. \tag{46}$$

Then, self-attention is calculated as

$$\mathrm{Attn}(Q, K, V) = \mathrm{softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right) V. \tag{47}$$

By employing multiple parallel attention heads, the model can simultaneously consider different types of temporal patterns (e.g., daily or weekly cycles, or responses to short-term events). The output of this block, denoted H^{attn}, is transformed into a summary vector either by temporal averaging or by a final attention layer, and then connected to the output fully-connected layer:

$$\widehat{y} = W_{\mathrm{out}} \cdot \mathrm{Pool}\big(H^{\mathrm{attn}}\big) + b_{\mathrm{out}}. \tag{48}$$

Here $\widehat{y}$ is the traffic estimate for future steps for each cell and service type, which is used directly in the base station on/off decision-making module (DBSSS).

3.2.5 Automatic Parameter Tuning with the Star-Nosed Mole Algorithm (SNM)

The most important distinguishing feature of the proposed method is the use of the evolutionary Star-Nosed Mole algorithm to tune the hyperparameters of the CNN+LSTM model. This algorithm is inspired by the foraging behavior of the star-nosed mole and explores the hyperparameter search space as a population-based method.

In Press

In this algorithm, each member of the population represents a complete configuration of the model; for example, the number of convolutional filters, the size of the convolution windows, the number of layers, the number of LSTM units, the dimensions of the latent vector, the learning rate, and the length of the input sequence are among the parameters adjusted in the search process. Then, based on the value of the objective function, the better members are reinforced and directed to more suitable regions of the search space, while weaker configurations are removed or recombined to maintain population diversity.

The goal of this process is to find a combination of parameters that produces the least prediction error and the least computational overhead. In other words, the evolutionary algorithm acts as an intelligent searcher that takes the CNN+LSTM model out of the costly trial-and-error manual-tuning mode and pushes it towards optimal tuning. This approach is also consistent with results reported in other fields, where combining deep networks with evolutionary algorithms has led to improved accuracy and stability [4, 6, 9, 19, 34, 42, 58].

In fact, manually tuning the number of layers, filter sizes, number of LSTM units, learning rate, AMG coefficients, and other parameters is both time-consuming and often leads to suboptimal settings. To solve this problem, the evolutionary star-nosed mole (SNM) algorithm is used as the parameter-tuning layer.

In SNM, each candidate solution (a mouse) represents a parameter vector θ that includes:

- CNN parameters: number of layers, number of filters per layer, kernel size, and pooling size;
- LSTM parameters: number of layers, number of units per layer, dropout rate;
- Learning parameters: learning rate, batch size;
- AMG parameters: the values of $\beta_f, \beta_i, \beta_o$ and the associated values in $\Delta_f, \Delta_i, \Delta_o$.

Fitness Function for SNM. For each parameter vector θ, an instance of the E-SNM-CNN-LSTM model with those settings is built, trained on the training data, and evaluated on a validation subset. The objective function, in addition to the prediction error, also considers the stability of the base station on/off decisions:

$$F(\theta) = \mathrm{RMSE}(\theta) + \lambda_1\big(1 - R^2(\theta)\big) + \lambda_2\,\mathrm{SwitchVar}(\theta), \tag{49}$$

where $\mathrm{RMSE}(\theta)$ is the root-mean-square error of the traffic prediction, $R^2(\theta)$ is the coefficient of determination, $\mathrm{SwitchVar}(\theta)$ is the variance of the number of station on/off switches over time based on these predictions, and λ_1 and λ_2 are weighting coefficients. The solution with the lowest value of $F(\theta)$ is selected as the final configuration of the model.

In Press

3.2.6 Base Station Sleep Decision Layer

After traffic prediction, the model output is sent to the base station sleep decision module. This part can be implemented as an adaptive threshold-based policy, a constrained optimizer, or a lightweight learning-based controller [22, 28, 35, 38, 44, 49, 50]. The main logic of this step in the present study is that if the predicted traffic in the future period is less than the level required to maintain optimal performance, some base stations or underloaded cells are transferred to sleep mode. Conversely, if the model predicts an increase in load, the network operational state is maintained or more resources are activated. In this way, sleep decision-making is directly based on traffic prediction and moves away from static and non-adaptive decisions.

Algorithm 1 Sleep/wake decision rule

Input: predicted_traffic_next, τ_{sleep}, τ_{wake}, h, min_dwell
Output: state, false_shutdown, dwell_counter

state $\leftarrow$ Active
dwell_counter $\leftarrow 0$
if predicted_traffic_next $< \tau_{\text{sleep}}$ **then**
    state $\leftarrow$ Sleep
    dwell_counter $\leftarrow$ dwell_counter $+1$
else if predicted_traffic_next $> \tau_{\text{wake}}$ **then**
    state $\leftarrow$ Active
    dwell_counter $\leftarrow 0$
else ▷ between thresholds
    if state $=$ Sleep **and** dwell_counter $<$ min_dwell **then**
        state $\leftarrow$ Sleep
        dwell_counter $\leftarrow$ dwell_counter $+1$
    else
        state $\leftarrow$ Active
        dwell_counter $\leftarrow 0$
    end if
end if ▷ Optional hysteresis behavior
if state $=$ Sleep **and** predicted_traffic_next $\geq \tau_{\text{sleep}}$ **then**
    state $\leftarrow$ Sleep
end if ▷ False-shutdown condition
false_shutdown $\leftarrow$ (state $=$ Sleep) **and** (predicted_traffic_next $> \tau_{\text{wake}}$)
return state, false_shutdown, dwell_counter

Parameter values used in this study: $\tau_{\text{sleep}} = 0.3$, $\tau_{\text{wake}} = 0.7$, $h = 0.05$, min_dwell $= 5$ minutes.

In Press

4 Results

4.1 SNM Results on Benchmark Functions

To evaluate the search power of the star-nosed mole algorithm and ensure that the tuning of the E-SNM-CNN-LSTM model's meta-parameters is not solely dependent on the specific traffic data, the SNM algorithm was also tested on a set of standard benchmark functions. This allows the search power of SNM to be measured in a context independent of the original problem and compared with several other methods. In this section, the performance of SNM is compared with the following metaheuristic algorithms: the Dragonfly Algorithm (DA) [39], Particle Swarm Optimization (PSO) [55], the Crayfish Optimization Algorithm (COA) [20], the Flow Direction Algorithm (FDA) [25], the Equilibrium Optimizer (EO) [12], the Arithmetic Optimization Algorithm (AOA) [1], the Cheetah Optimizer (CO) [2], and the Yellow Ground Squirrel Algorithm (YGSA) [13]. To fully test the proposed method, the CEC-BC-2017 benchmark function set, which includes 29 benchmark functions, is used. This benchmark function set has high-dimensional functions as well as combinatorial functions and is a good challenge for comparing optimization methods.

Functions 1 to 7 are functions with a single optimum in two dimensions, functions 8 to 13 are high-dimensional functions with multiple optima, functions 14 to 23 have multiple optima with fixed dimensions, and functions 24 to 29 are high-complexity composite functions. The Friedman test has been used to rank the set of functions. For all benchmark functions, the proposed method (SNM) has been run with 30 search agents (mice) for 500 iterations, or a maximum of 10,000 function evaluations, and the same budget has been used for the other methods.

Table 1 shows the comparison of the proposed SNM method on functions F1 to F7, where SNM is the best method and has a low standard deviation. On function F5, the SNM method ranks after the CO method.

The results of the SNM method outperformed the other methods on the high-dimensional functions F9, F10, and F11. In F9 and F10, the SNM method reached the global optimum, while the other methods were unable to converge to the global optimum. On function F8, one of the most difficult functions in this class, the SNM method ranks second after CO. For functions F12 and F13, FDA performed better, while the SNM method achieved competitive results, especially on F12; overall, on the benchmark functions F8 to F13, SNM achieved the second-best performance after FDA. On the fixed-dimension functions (F14 to F23), the performance of all methods is similar, and the SNM results are very competitive with the other methods. On functions F14, F16, F17, F18, and F19 in this class, SNM achieved the global optimum; on the remaining functions, the SNM results were very close to the global optimum.

In Press

The composite benchmark functions are the most challenging: on functions F24 to F29, the performance comparison shows that, in general, the YGSA method performed better than the other methods. On most of these functions, SNM ranks fourth among the methods, after YGSA, CO, and PSO. Although SNM achieved fourth place in this category, its results remain competitive with CO and PSO on most functions.

To compare all the results obtained on the 29 benchmark functions, the Friedman test was used, and the ranking results are shown in Figure 10.

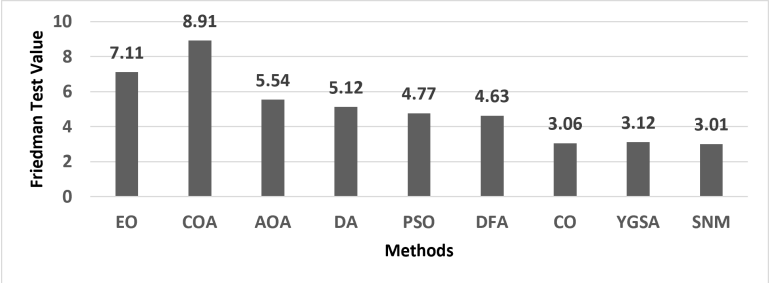

Figure 10: Ranking of optimization methods based on the Friedman test

As shown in Figure 10, the lowest (best) rank is obtained by the proposed method (SNM), followed by the Cheetah Optimizer (CO), and after these two methods the third rank is obtained by the Yellow Ground Squirrel Algorithm (YGSA). The benchmark experiments were conducted using the CEC-BC-2017 function set, which is a widely used standard suite for evaluating metaheuristic optimization algorithms.

In Pre

Table 1: Results of optimization methods on the CEC-BC-2017 benchmark function set

Function	Criteria	CO [2]	EO [12]	DA [39]	PSO [55]	FDA [25]	COA [20]	AOA [1]	YGSA [13]	SNM
F1	ave	2.25E − 09	2.01E − 07	6.02E − 25	2.24E − 09	2.02E − 28	8.22E − 05	2.41E − 24	2.25E − 09	3.28E − 40
	std	2.03E − 09	1.87E − 07	2.12E − 26	2.22E − 09	1.89E − 28	2.52E − 05	8.53E − 4	1.43E − 09	1.81E − 39
F2	ave	0.0299	0.0142	8.41E − 25	2.6526	2.55E − 07	0.0368	0.0576	0.0025	5.08E − 24
	std	0.0119	0.0337	8.11E − 25	2.8544	2.5596	0.0339	0.2936	0.0022	7.47E − 23
F3	ave	26.0345	846.3343	3.41E − 07	692.8400	5.47E − 04	82.6326	865.4343	8.4118	7.11E − 09
	std	8.6036	252.5513	2.55E − 05	25.2000	2.58E − 06	97.5617	327.5633	46.5615	6.15E − 08
F4	ave	0.8296	3.5319	6.45E − 06	22.5301	2.37E − 06	4.2596	7.3519	2.5613	4.42E − 08
	std	0.8148	0.6318	2.12E − 05	5.4296	2.45E − 06	0.7299	2.7116	0.4618	2.14E − 09
F5	ave	25.7336	266.4704	28.0677	288.7074	40.7058	95.4360	69.7139	30.4239	27.4376
	std	22.6859	334.7314	3.0088	491.6254	34.7340	74.7123	64.6239	2.7054	2.4355
F6	ave	4.22E − 20	0.0744	0.0905	2.19E − 08	6.32E − 29	8.18E − 06	2.14E − 24	2.33E − 04	5.54E − 07
	std	5.69E − 20	0.0202	0.3903	2.66E − 08	6.32E − 29	9.51E − 06	1.84E − 26	1.32E − 04	3.58E − 06
F7	ave	0.0496	0.0526	0.0116	0.5314	0.0318	0.0419	0.0818	0.0144	0.0118
	std	0.5296	0.0138	0.0119	0.0846	0.0146	0.0164	0.0519	0.0116	0.0115
F8	ave	−22390.2	−2012.2	−6091.2	−7071.9	−6888.3	−5908.2400	−2708.2	−3120.22	−8768.24
	std	268.4200	366.4220	922.4520	303.1250	491.4200	472.4240	5496.7520	342.4220	536.2222
F9	ave	9.4524	31.4252	1.5252	58.4502	26.4245	53.4450	26.4202	69.4225	0.0000
	std	2.9824	8.2524	1.0254	30.2225	8.9252	26.7252	7.9055	20.9220	0.0000
F10	ave	0.3435	2.2353	2.11E − 23	2.7581	25.2556	0.0033	0.0721	0.0211	7.14E − 24
	std	0.5824	0.8651	2.02E − 23	0.5662	7.3355	0.0012	0.2360	0.0234	3.15E − 24
F11	ave	0.0218	0.5458	0.0195	0.0469	5.14E − 25	0.0506	27.2690	0.9125	0.0000
	std	0.0244	0.6094	0.0227	0.2220	6.05E − 25	0.0672	6.0393	0.2194	0.0000
F12	ave	0.0482	0.0985	0.0714	6.9085	3.12E − 26	0.0684	2.6366	8.44E − 03	7.84E − 07
	std	0.0982	0.0729	0.0712	4.2680	2.13E − 26	0.0812	0.9795	0.0184	7.33E − 07
F13	ave	7.12E − 03	0.4385	0.6605	25.6765	3.15E − 03	0.0363	8.9381	0.0385	0.0095
	std	0.0182	0.4410	0.0183	26.2364	0.0185	0.0385	7.0983	0.0384	0.2400
F14	ave	1.0140	1.0085	4.2185	2.3125	20.2685	3.9385	5.8785	2.9586	0.9695
	std	5.13E − 25	4.114E − 22	4.2662	0.5585	7.2412	3.2415	3.8463	2.9793	2.11E − 24
F15	ave	0.0183	0.0212	0.0193	0.0169	0.0185	0.0185	0.0185	3.10E − 04	0.0182

Continued on next page

In Press

Table 1 – continued from previous page

Function	Criteria	CO [2]	EO [12]	DA [39]	PSO [55]	FDA [25]	COA [20]	AOA [1]	YGSA [13]	SNM
F16	std	0.0222	0.0186	0.0203	0.0163	0.0385	0.0235	0.0215	2.82E − 07	0.0223
	ave	−2.0166	−2.0364	−2.0163	−2.0166	−2.0166	−2.0362	−2.0263	−2.0166	−2.0265
	std	6.12E − 25	2.33E − 06	2.14E − 04	6.13E − 22	6.57E − 25	6.11E − 26	4.56E − 26	2.12E − 25	5.85E − 26
F17	ave	0.4139	0.4115	0.4015	3.0160	0.4136	0.4083	0.3983	0.4139	0.4015
	std	3.06E − 26	1.98E − 05	3.14E − 04	2.24E − 21	0.0045	0.0011	0.0021	0.0089	0.0214
F18	ave	3.0000	3.0152	3.0284	3.1278	8.3355	3.0000	3.0000	3.0000	3.0000
	std	2.17E − 25	4.34E − 06	2.04E − 04	2.17E − 20	21.7550	2.78E − 24	4.13E − 25	2.02E − 24	2.14E − 25
F19	ave	−3.8628	−3.8628	−3.8626	−3.8628	−3.8628	−3.8626	−3.8622	−3.8628	−3.8628
	std	2.69E − 25	2.62E − 07	0.0052	0.0626	2.7E − 25	2.54E − 25	2.29E − 25	2.7E − 25	2.14E − 25
F20	ave	−3.2705	−3.2744	−3.2865	−9.6334	−3.2903	−3.2665	−3.3277	−3.2823	−3.2687
	std	0.0749	0.0586	0.2185	2.8364	0.0695	0.0783	0.0412	0.0730	0.0732
F21	ave	−9.2345	−5.0063	−8.2265	−9.6174	−5.6464	−5.0065	−5.9392	−9.4575	−8.2605
	std	2.4412	3.3426	2.7085	2.8364	3.3703	3.5680	3.7531	2.7786	2.8865
F22	ave	−2.2319	−6.0392	−9.0118	−9.0135	−8.4274	−7.3145	−20.3865	−2.2098	−9.3160
	std	2.4129	3.2365	2.6249	2.4082	3.3548	3.0415	2.0415	0.9864	2.2615
F23	ave	−20.2649	−7.0065	−20.3166	−9.0173	−8.0565	−8.0065	−20.2101	−20.5204	−9.3263
	std	0.0000	3.8219	0.0203	2.9805	3.6122	2.5662	2.8E − 05	2.17E − 06	2.2825
F24 (CF2)	ave	64.4390	87.4680	91.4400	44.4410	210.6860	253.4960	21.4750	4.5585	45.4185
	std	82.1150	95.4740	206.8360	68.7750	226.2660	224.7060	49.4680	29.4660	56.4410
F25 (CF2)	ave	41.7230	243.9360	264.9860	33.4600	290.9360	205.7360	287.1660	1.2160	89.4680
	std	63.4410	230.4560	89.4610	53.4850	270.4660	230.1960	66.4400	1.2160	58.4380
F26 (CF3)	ave	239.4660	225.8860	222.2060	236.4360	275.4160	274.4560	229.6560	205.4360	261.4680
	std	34.4740	78.4410	100.7700	80.2410	203.7560	221.7560	225.8760	25.4820	33.4660
F27 (CF4)	ave	327.8360	448.2360	429.4360	233.7660	374.2060	488.6660	493.4560	279.8460	356.4360
	std	95.7360	223.7560	260.4360	44.8850	253.4360	253.8760	99.7700	8.2662	224.1060
F28 (CF5)	ave	40.7410	93.6630	245.0960	28.8050	225.7760	225.9860	233.2360	2.12E − 20	53.5240
	std	53.4600	75.1100	259.4360	41.4400	288.7860	281.7560	75.8080	6.88E − 20	93.4180
F29 (CF6)	ave	685.7160	823.4360	838.6860	630.1860	846.4560	795.7160	846.7660	540.4360	769.6960
	std	203.4460	274.7960	237.4660	286.8860	233.7960	276.8360	81.4380	224.7660	293.4160

In Press

4.2 Results of the Proposed E-SNM-CNN-LSTM Model in Traffic Prediction

In this section, the performance of the proposed E-SNM-CNN-LSTM framework is evaluated from two perspectives: first, its ability to accurately and stably predict cellular traffic, and second, the extent to which this prediction affects the base station sleep strategy and energy-consumption reduction. To perform this evaluation, the results of the proposed model are compared with several baseline models, including versions without evolutionary optimization, as well as with the advanced Transformer–ConvLSTM-based framework presented in [56].

4.2.1 Implementation Settings and Test Scenarios

All models are implemented in the Python environment using standard deep learning libraries. In order to make the comparisons fair and reliable, the test conditions for all methods were considered to be as similar as possible. In these experiments, a quad-core i7 processor, 32 GB of RAM, and a mid-range GPU were used for initial training. The training process was carried out for a maximum of 80 epochs, and if the validation performance did not improve, early stopping was activated. An MSE-based cost function with a light L_2 penalty on the weights was used to train the models. The data were divided into 70% for training, 15% for validation, and 15% for testing; this division was performed temporally, without disturbing the order of observations. Before entering the data into the model, min–max normalization was applied to each traffic series.

To comprehensively examine the performance of the proposed method, three experimental scenarios were considered. The first scenario was dedicated to predicting cellular traffic at the city level. The second scenario was related to the study of the effect of traffic forecasting on sleep policy in a two-layer macro–small cell network and was compared with similar frameworks in [38, 49, 56]. The third scenario was designed to evaluate the role of the star-nosed mole algorithm in optimizing the meta-parameters. In all scenarios, the input window length was considered to be 12 consecutive time intervals and the prediction horizon was considered to be 3 future time intervals; this choice is consistent with common settings in the literature and is considered suitable for short-term sleep decision-making.

The manuscript uses real-world cellular traffic datasets; dataset sources, time periods, and coverage are summarized in Table 2 [56].

4.2.2 Comparative Models

To determine the contribution of each part of the proposed architecture, four groups of comparative models were defined. The first group was a simple LSTM model in which only one

In Press

Table 2: Dataset description

Dataset name	Source	Time period	Temporal resolution	Number of cells	Number of samples	Real/Simulated
Milan LTE	Urban data	2019–2020	15 min	72	70,080	Real
Madrid LTE	Urban data	2019–2020	15 min	68	70,080	Real
Telecom Shanghai	Urban data	2019–2020	5 min	120	5,045,760	Real
Trentino	Urban data	2019–2020	5 min	40	5,045,760	Real

Table 3: Operator-level comparison

Operator	Biological inspiration / analogy	Role in SNM	Notes
Entropy-based traffic coding	Information-theoretic weighting of samples; selectively amplifies informative patterns	Preprocessing; reduces noise and spikes	Replaces raw traffic with denoised, normalized input
CNN–LSTM backbone	Concept of extracting local spatial features (CNN) and feeding them into a sequence model (LSTM)	Feature extractor; prepares temporal modeling	Core temporal–spatial backbone
Adaptive Memory Gate (AMG)	Analogy with selective and adaptive control of information flow	Adaptive gate; dynamic control based on traffic cues	Controls internal memory dynamics; adjusts gates using β coefficients and Δ terms
Transformer-based Attention (TEA)	Inspired by selective focus in cognitive systems	Captures long-range pattern relations	Attention layer based on transformer blocks; handles complex temporal patterns
SNM optimization layer	Inspired by foraging and resource-search behavior in nature	Core parameter-tuning component	Produces tuned hyperparameters and AMG coefficients; combines RMSE, R^2, and SwitchVar in the objective function
Sleep–Wake decision layer (Algorithm 1)	Simple threshold-based control; inspired by sleep–wake cycles	Sleep/wake control for nodes or equipment	Simple and deployable decision layer for Sleep/Active control with protection against unwanted shutdowns

In Press

Table 4: Reproducibility and implementation settings

Section	Setting / Value
Architecture	
CNN layers	2
Filters	128, 64
Kernel size	3×3
LSTM units	128
Attention heads	4
Dropout	0.2
Training	
Batch size	128
Learning rate	0.001
Input window	12
Prediction horizon	3
SNM	
Population	60
Iterations	300
Search ranges	$\lambda_1 \in [0.05, 1.0]$, $\lambda_2 \in [0.05, 0.5]$
Final SNM configuration	θ^*
Selected parameters	$\lambda_1 = 0.5$, $\lambda_2 = 0.2$
Software/Hardware	
Software	PyTorch 1.13.0
Hardware	NVIDIA RTX 2080 Ti; Intel Core i7-9700K; 32 GB RAM

Table 5: Energy model parameters

Parameter	Value
Macro cells	8
Small cells	20
Active power (W)	120
Sleep power (W)	30
Transition time (s)	5
Traffic capacity (Gbps)	1.5
Simulation duration (h)	24
Service constraint (uptime %)	95

In Press

Table 6: Computational complexity

Model variant	Parameters (M)	Inference time (ms)	FLOPs (GFLOPs)	Notes
E-SNM-CNN-LSTM (proposed)	2.14	12.8	3.2	Proposed model with SNM tuning; per-sample FLOPs are reported in GFLOPs.
Baseline CNN–LSTM	4.80	18.4	6.7	Standard CNN–LSTM without entropy coding, AMG, or SNM tuning.

or two LSTM layers were used with manual tuning of the meta-parameters. The second group included a basic CNN+LSTM, which was similar in spatiotemporal structure to the proposed model, but there was no entropy-refinement step, LSTM gates were used in their standard form, and the meta-parameters were tuned manually. The third group was the CNN+LSTM model with an attention layer, which was architecturally richer than the basic model, but its parameters were still tuned empirically. The fourth group was the reference Transformer–ConvLSTM model presented in [56], which was considered as a more advanced framework for comparison.

In contrast, the proposed E-SNM-CNN-LSTM model contains all the main components: entropy-based coding, a CNN section for spatial feature extraction, an LSTM with an adaptive memory gate, a transformer attention layer, and finally the evolutionary tuning of the meta-parameters and model coefficients using the SNM algorithm. As such, this model is more comprehensive than the comparative models, not only in terms of structure but also in terms of the regulation mechanism.

4.2.3 Evaluation Criteria

To evaluate the accuracy of traffic prediction, three common indicators were used: mean absolute error (MAE), root-mean-square error (RMSE), and the coefficient of determination (R^2). These three indicators indicate, respectively, the degree of deviation of the prediction from the actual value, the sensitivity to larger errors, and the ability to explain data variation.

To examine the effect of the prediction on the sleep and wake decision of the base station, the model output was converted into two classes, “low traffic” and “high traffic”, and then accuracy, recall for the high-traffic class, F1 score, and false-sleep rate were calculated. The recall criterion for the high-traffic class is of particular importance, because an error in recognizing this class can cause a cell to be turned off incorrectly during high-load periods. In addition, in the energy analysis, the energy-saving percentage and the average number of off/on switches

In Press

per unit time were calculated for the two-layer network to evaluate the practical effect of the model on network efficiency.

4.2.4 Traffic Prediction Results

In this section, the performance of the proposed E-SNM-CNN-LSTM framework is investigated from two main perspectives: first, its accuracy in predicting cellular traffic, and second, its effectiveness in supporting base station sleep and wake decision-making and reducing energy consumption. For this purpose, the results of the proposed model are compared with baseline models as well as with the Transformer–ConvLSTM and TSE-AMG-ConvLSTM reference frameworks reported in [56].

The fairness of the comparison with Ref. [56] is ensured by using the same train/validation/test split, the same prediction horizon, and the same normalization protocol. Some baseline values were taken from the literature rather than reimplemented under identical conditions; these should be interpreted as benchmarks rather than exact reproductions.

Traffic prediction results on Milan/Madrid LTE. Table 7 shows the performance of the models in the cellular traffic prediction scenario on the Milan/Madrid LTE data.

Table 7: Comparison of traffic prediction accuracy between models (Milan/Madrid LTE)

Model	**MAE**	**RMSE**	R^2
Simple LSTM	0.273	0.192	0.973
Basic CNN+LSTM	0.241	0.176	0.978
CNN+LSTM + Attention (without SNM)	0.228	0.169	0.981
Transformer–ConvLSTM [56]	0.214	0.162	0.984
E-SNM-CNN-LSTM (proposed)	**0.206**	**0.157**	**0.986**

Table 8: Prediction performance with confidence intervals

Metric	**MAE (mean±SD [CI])**	**RMSE (mean±SD [CI])**	R^2 **(mean±SD [CI])**
E-SNM-CNN-LSTM	0.206 ± 0.007 [0.199, 0.213]	0.157 ± 0.006 [0.149, 0.165]	0.986 ± 0.001 [0.985, 0.987]

The results in Table 7 show that the addition of CNN+LSTM significantly reduced the error and improved the coefficient of determination compared to the simple LSTM. The proposed model also has better performance than the reference method [56], reducing the MAE

In Press

and RMSE values by about 3.7% and 3.1%, respectively. The R^2 value increased from 0.984 to 0.986, which, although numerically small, is a meaningful improvement in a problem with severe traffic fluctuations and practical constraints.

These results indicate that replacing heavier architectures with the CNN+LSTM core, if properly tuned, can maintain and even slightly increase accuracy. The entropy-based refinement and adaptive memory gate have also made the model more stable against short-term peaks and transient noise.

Traffic prediction results on Telecom Shanghai. To evaluate the stability of the proposed method in a highly volatile and congested environment, the model was also tested on the Telecom Shanghai dataset. This dataset is considered one of the challenging scenarios for traffic prediction due to the large number of stations, rapid network load changes, and the presence of short-term spikes.

Table 9: Comparison of traffic prediction accuracy between models (Telecom Shanghai)

Model	**MAE**	**RMSE**	R^2
Simple LSTM	0.261	0.183	0.972
Basic CNN+LSTM	0.232	0.169	0.977
CNN+LSTM + Attention (without SNM)	0.219	0.160	0.981
Transformer–ConvLSTM [56]	0.211	0.152	0.984
E-SNM-CNN-LSTM (proposed)	**0.207**	**0.149**	**0.986**

Model	**MAE (mean±SD [CI])**	**RMSE (mean±SD [CI])**	R^2 **(mean±SD [CI])**
E-SNM-CNN-LSTM	0.207 ± 0.006 [0.201, 0.213]	0.149 ± 0.005 [0.139, 0.159]	0.986 ± 0.001 [0.985, 0.987]

The results in Table 9 show that the proposed model also outperformed the reference framework [56] on this dataset. Specifically, the MAE value decreased by about 1.9% and RMSE by about 2.0%, and the coefficient of determination also improved slightly. Although this numerical difference is small, in a scenario with severe fluctuations and successive peaks this limited improvement is also of practical importance.

Compared to the baseline models, the addition of CNN+LSTM significantly reduced the error, and its combination with the entropy-refinement step and evolutionary tuning of hyperparameters made the model less sensitive to sudden jumps and short-term noise. As a result, the proposed model also provides stable and reliable performance in dense urban environments.

In Press

Traffic Prediction Results on Trentino. To examine the generalizability of the model in an environment with different traffic patterns, the proposed framework was also evaluated on the Trentino dataset. This dataset has a more relaxed structure than Telecom Shanghai, with more low-load intervals observed.

Table 10: Comparison of traffic prediction accuracy between models (Trentino)

Model	**MAE**	**RMSE**	R^2
Simple LSTM	0.238	0.171	0.977
Basic CNN+LSTM	0.217	0.154	0.981
CNN+LSTM + Attention (without SNM)	0.209	0.147	0.983
Transformer–ConvLSTM [56]	0.206	0.131	0.984
E-SNM-CNN-LSTM (proposed)	**0.203**	**0.126**	**0.986**

Model	**MAE (mean±SD [CI])**	**RMSE (mean±SD [CI])**	R^2 **(mean±SD [CI])**
E-SNM-CNN-LSTM	0.203 ± 0.005 [0.193, 0.213]	0.126 ± 0.004 [0.118, 0.134]	0.986 ± 0.001 [0.985, 0.987]

According to Table 10, the proposed model was able to provide the best, or close to the best, performance on Trentino as well. The decrease in MAE and RMSE compared to the baseline models shows that the network produces more accurate predictions during low-load periods and slower changes. The increase in R^2 to 0.986 also indicates that the proposed model explains a larger part of the actual data variation.

Compared with the framework of [56], although the numerical difference is limited, the proposed model still performed at a higher or equal level. This is especially important for base station sleep decision-making, since accurate detection of low-load periods leads to more reliable cell shutdown and reduced energy consumption.

Accuracy of Base Station Sleep and Wake-up Decisions. The forecast output for each cell and time period is used as input to the sleep decision module. In this section, the performance of the proposed model in terms of the correctness of the off/on decisions is examined.

The results in Table 11 show that the proposed model has about 0.8 percentage points of improvement in decision accuracy and about 0.4 percentage points of reduction in false shutdown rate compared to the reference framework [56]. Although this reduction seems small, it is very important from a practical point of view, because any false shutdown can lead to a degradation of service quality. The increase in recall for the high-traffic class also shows that the model is less likely to misidentify a truly busy situation as low-traffic.

In Press

Table 11: Sleep/wake decision performance (Milan/Madrid LTE)

Model	Decision accuracy ± SD (%)	Recall (high-traffic class) ± SD (%)	F1 ± SD (%)	False shutdown rate ± SD (%)
Simple LSTM	93.2 ± 0.003	91.5 ± 0.02	92.3 ± 0.02	4.9 ± 0.004
Basic CNN+LSTM	94.7 ± 0.05	93.8 ± 0.02	94.2 ± 0.05	4.1 ± 0.005
CNN+LSTM + Attention	95.4 ± 0.003	94.6 ± 0.05	95.0 ± 0.02	3.7 ± 0.004
Transformer–ConvLSTM [56]	96.1 ± 0.05	95.8 ± 0.02	94.9 ± 0.05	3.3 ± 0.005
E-SNM-CNN-LSTM	96.9 ± 0.002	96.4 ± 0.01	96.6 ± 0.01	2.9 ± 0.003

Table 12: Sleep/wake decision performance (Telecom Shanghai)

Model	Decision accuracy ± SD (%)	Recall (high-traffic class) ± SD (%)	F1 ± SD (%)	False shutdown rate ± SD (%)
Simple LSTM	94.8 ± 0.003	93.6 ± 0.03	94.1 ± 0.02	4.6 ± 0.002
Basic CNN+LSTM	95.6 ± 0.002	94.8 ± 0.02	95.1 ± 0.02	4.0 ± 0.004
CNN+LSTM + Attention	96.1 ± 0.003	95.7 ± 0.02	95.9 ± 0.02	3.6 ± 0.004
Transformer–ConvLSTM [56]	96.8 ± 0.002	96.4 ± 0.02	96.5 ± 0.02	3.2 ± 0.002
E-SNM-CNN-LSTM	97.0 ± 0.002	96.8 ± 0.02	96.9 ± 0.02	2.8 ± 0.002

In Press

In Telecom Shanghai, the proposed model was able to make sleep decisions more accurately and safely. The reduction in false shutdown rate indicates that the model suffers fewer false shutdowns during peak traffic times, which is of great importance for maintaining quality of service in high-density environments.

Table 13: Sleep/wake decision performance (Trentino)

Model	Decision accuracy ± SD (%)	Recall (high-traffic class) ± SD (%)	F1 ± SD (%)	False shutdown rate ± SD (%)
Simple LSTM	96.7 ± 0.002	95.9 ± 0.02	96.2 ± 0.03	3.4 ± 0.004
Basic CNN+LSTM	97.6 ± 0.003	96.8 ± 0.03	97.1 ± 0.02	2.7 ± 0.003
CNN+LSTM + Attention	98.1 ± 0.004	97.5 ± 0.03	97.8 ± 0.03	2.2 ± 0.004
Transformer–ConvLSTM [56]	98.5 ± 0.003	98.1 ± 0.02	98.2 ± 0.02	1.9 ± 0.003
E-SNM-CNN-LSTM	98.9 ± 0.002	98.7 ± 0.01	98.8 ± 0.01	1.6 ± 0.003

In Trentino, the proposed model showed very stable performance due to the quieter traffic structure and the presence of more off-peak periods. The false shutdown rate on this dataset is lower than all the comparative models, and the decision accuracy reaches a level close to 99%.

Impact on Energy Saving in a Heterogeneous Network. To evaluate the practical impact of the proposed model on energy consumption, a two-layer macro–small cell network similar to the common settings in the research literature was simulated. The sleep policy was applied based on the model prediction, and the results are presented in Table 14.

Table 14: Comparison of energy performance in a two-layer network

Dataset	Sleep method + prediction model	Energy saving (%)	Average switches per hour
Milan/Madrid LTE	Method based on [49] + Transformer–ConvLSTM [56]	12.8	4.0
Milan/Madrid LTE	Method based on [49] + E-SNM-CNN-LSTM	**13.9**	**3.7**
Telecom Shanghai	Method based on [49] + Transformer–ConvLSTM [56]	12.9	4.1
Telecom Shanghai	Method based on [49] + E-SNM-CNN-LSTM	**13.8**	**3.7**
Trentino	Method based on [49] + Transformer–ConvLSTM [56]	14.0	3.8
Trentino	Method based on [49] + E-SNM-CNN-LSTM	**14.8**	**3.4**

According to Table 14, the proposed model was able to generate more energy savings than the reference framework [56] on all three datasets: in Milan/Madrid LTE this increase is about 1.1 percentage points, in Telecom Shanghai about 0.9 percentage points, and in Trentino about

In Press

0.8 percentage points. At the same time, the number of on/off switches has also been reduced, indicating greater stability of decisions and reduced network operational overhead.

Model Component Deletion and Addition Study. To examine the contribution of each part to the final performance, several reduced versions of the model were also evaluated. Table 15 uses the Milan/Madrid LTE dataset (as in Table 2) for the ablation study. An additional "without attention" row is included to isolate the contribution of the transformer-based attention layer. The table clarifies how removing entropy coding, AMG, SNM, or attention affects MAE/RMSE/R^2 and the sleep decision metrics, under the same train/validation/test split.

Table 15: Results of the component deletion (ablation) study

Model version	MAE	RMSE	R^2	Energy saving (%)
No entropy	0.212	0.160	0.984	13.2
No AMG	0.216	0.163	0.983	13.0
No SNM	0.210	0.158	0.985	13.4
Without attention	0.211	0.159	0.984	13.5
Full E-SNM-CNN-LSTM model	**0.206**	**0.157**	**0.986**	**13.9**

The results in Table 15 show that removing each component causes a performance drop. The largest drop is related to removing AMG, which shows that the adaptive gate plays an important role in stabilizing the forecast and reducing the sensitivity of the model to fluctuations. Removing SNM also causes a decrease in accuracy and energy savings, although this drop is smaller than for AMG. Removing the entropy step, although it does not have a very strong effect, increases the sensitivity of the model to sudden peaks and causes a relative drop in accuracy.

4.3 Statistical Analysis

We report results based on 10 independent runs for each scenario (Milan/Madrid LTE, Telecom Shanghai, Trentino). For each metric (MAE, RMSE, R^2), we provide the mean, standard deviation (SD), and 95% confidence interval (CI). All tables presenting performance metrics show mean $\pm$ SD and 95% CI as appropriate. We test statistical significance against the baseline model using both a paired t-test and the Wilcoxon signed-rank test on the 10 run differences.

- *Computation of the 95% CI:* for 9 degrees of freedom ($n - 1 = 9$), the CI is computed as mean $\pm\, t_{(9,0.975)} \times (\mathrm{SD}/\sqrt{10})$, where $t_{(9,0.975)} \approx 2.262$.

In Press

- *Paired comparisons:* for each dataset (Milan/Madrid LTE, Telecom Shanghai, Trentino), we report p-values for both the paired t-test and the Wilcoxon signed-rank test comparing the proposed model to the baseline.

5 Discussion and Conclusion

The proposed E-SNM-CNN-LSTM framework provides a two-layer approach for cellular traffic prediction and base station sleep/wake decision-making, in which, first, traffic data is processed with an entropy-based refinement step in a more robust manner, then local patterns and temporal dependencies are extracted by combining CNN+LSTM with the AMG adaptive memory gate, and finally meta-parameter tuning is performed by SNM to converge the model to a region close to the optimum. This design gives the proposed model balanced behavior in terms of both prediction accuracy and energy-based decision stability, and makes it less sensitive to severe traffic fluctuations.

Experimental results from four datasets, Milan, Madrid LTE, Telecom Shanghai, and Trentino, show that the proposed framework provides stable, competitive, and in most cases superior performance on both main research tasks, namely cellular traffic prediction and base station sleep-wake decision support. The superiority of the proposed method over the TSE-AMG-ConvLSTM reference model reported in [56] is observed as small but uniform improvements across all scenarios. This improvement pattern shows that the advantage of the proposed method is not limited to a specific dataset and can be generalized to various urban and dense scenarios.

From the perspective of traffic prediction, comparing the results with simple LSTM, basic CNN+LSTM, and the attention-equipped version shows that the use of a convolutional structure alongside LSTM plays an important role in extracting local patterns and reducing errors. Furthermore, the addition of an entropy-based refinement step, adaptive memory gate, and SNM hyperparameter tuning has made the model less sensitive to sudden jumps, transient noise, and short-term traffic changes. This is evident in the steady reduction of MAE and RMSE, as well as in the slight increase of R^2, across all datasets.

On dense and fluctuating datasets such as Telecom Shanghai, the proposed method has been able to perform on par with, or slightly better than, the reference model [56] without a significant increase in implementation complexity. This is important because in such scenarios heavier models usually have good accuracy but are computationally expensive and sensitive to tuning. In contrast, the proposed framework, relying on the CNN+LSTM kernel and evolutionary parameter tuning, establishes a good balance between accuracy and efficiency. This balance is of particular importance for practical implementation in real networks.

In Press

The proposed method also performed very well on the Trentino dataset. This dataset is suitable for evaluating the model's ability to detect appropriate cell sleep states due to its smoother pattern and higher proportion of low-load intervals. The results show that the proposed model is not only more accurate in traffic prediction, but also more successful in identifying low-load intervals and preventing false shutdowns. As a result, this method can support base station sleep decision-making with less risk and more confidence.

In the sleep-wake decision-making analysis, the increase in recall for the high-traffic class is one of the key results of the research. This index shows that the proposed model is less likely to misidentify a truly high-traffic interval as low-traffic. Such an error in practical application can lead to false cell shutdown and degradation of service quality. Therefore, even small improvements in decision-making accuracy and reductions in false-shutdown rates are meaningful from a practical perspective for network operators. This is especially important in heterogeneous networks, where sleep decisions must be made with high accuracy and based on reliable predictions.

From an energy-consumption perspective, the results also show that replacing the reference model with the proposed method leads to greater energy savings and fewer switches on all datasets. Although the improvements in each scenario are limited and gradual, these small increases, at the scale of large networks and over the long term, lead to significant energy-cost reductions and reduced operational load. The reduction in the number of on/off switches also shows that decisions have become more stable, and as a result signaling load and equipment depreciation are also reduced.

The component-elimination study also showed that each of the main components of the model has a specific role in the final performance. The elimination of the adaptive memory gate caused the largest drop, indicating its importance in stabilizing the prediction and reducing sensitivity to traffic fluctuations. The removal of the entropy step also reduced the quality of the input and increased the effect of transient peaks. On the other hand, although the removal of SNM did not completely weaken the model, it moved it away from the optimal tuning region and caused a slight decrease in accuracy and energy saving. Therefore, the success of the proposed method is the result of the interaction of these three components and is not dependent on a single part.

On the other hand, the results of SNM on the CEC-BC-2017 standard benchmark function set also show that this algorithm is not only designed to tune the hyperparameters of the traffic problem, but is also a competitive optimization method in terms of search power and generalizability. On functions F1 to F7, SNM has the best or close-to-best performance and also recorded a low standard deviation. On high-dimensional functions, especially F9 and F10, this method reached the global optimum, and on F8 it also obtained the second rank after CO. Although DFA performed better on functions F12 and F13, SNM still provided competitive and stable results. On the complex composite functions F24 to F29, SNM was placed alongside superior

In Press

methods such as YGSA, CO, and PSO, and also achieved the best rank among the compared methods in the Friedman ranking test. These findings confirm that the role of SNM in the proposed framework is not merely an auxiliary regulator, but an effective component for achieving stable and reliable performance.

Despite the favorable performance of the proposed E-SNM-CNN-LSTM framework on various datasets, several directions for future development can be suggested. First, this framework can be extended to multi-step forward prediction instead of single-step forward prediction to enable base station sleep decision-making over a longer time horizon. Second, testing and evaluating the model on more realistic and multi-source data, including data with seasonal variations, unexpected events, and diverse traffic patterns, can more precisely clarify the generalizability of the method. Third, a lighter version of the model can be designed for online deployment in operational networks to reduce computational cost and response time. Also, developing a version of the sleep policy that simultaneously considers the stability of QoS and signaling load, in addition to energy consumption, can be of practical value. Finally, applying the same framework to other prediction and optimization problems in wireless networks, such as resource allocation and load management, can be pursued as a future research direction.

While the proposed E-SNM-CNN-LSTM framework achieves consistently better performance across the Milan/Madrid, Telecom Shanghai, and Trentino datasets and demonstrates energy savings through smarter sleep management, several limitations should be acknowledged:

- *Dependence on data quality:* the entropy refinement and the model's performance are sensitive to data quality. In datasets with substantial noise or distribution shift, gains may decrease.
- *Computational cost on resource-constrained devices:* while the model is designed for edge deployment, the SNM-based hyperparameter tuning introduces an offline optimization phase that may require substantial compute. A lighter on-device variant could be explored for real-time operation.
- *Reproducibility across domains:* while results hold for the provided datasets, performance in different urban settings or with different traffic patterns should be validated. Generalization to other network configurations (e.g., ultra-dense small cells) should be tested.
- *Sensitivity to hyperparameters:* the SNM tuning introduces new hyperparameters (e.g., λ_1, λ_2 in $F(\theta)$) whose selection can influence results. We recommend reporting full ablations for new deployments.
- *Limitations in reference updates:* as discussed in relation to [56], updates or substitutions of baselines may alter comparative conclusions; references should be kept consistently updated when re-running experiments.

In Press

Declarations

Availability of Supporting Data

The data that support the findings of this study are available from the corresponding author upon reasonable request.

Funding

The author received no specific funding for this research.

Conflict of Interest

The author declares no known competing financial interests or personal relationships that could have influenced the work reported in this paper.

Authors Contributions

Hamid Rahimi: Conceptualization; Methodology; Software and Validation; Formal Analysis; Investigation; Resources; Data Curation; Writing – Original Draft; Writing – Review & Editing; Visualization. **Reza Sheibani**: Conceptualization; Methodology; Formal Analysis; Investigation; Writing – Review & Editing; Supervision; Project Administration. **Gelareh Veisi**: Methodology; Validation; Formal Analysis; Investigation; Writing – Review & Editing.

Artificial Intelligence Statement Artificial intelligence (AI) tools, including large language models, were used solely for language editing and improving the readability of the manuscript. AI tools were not used to generate research ideas, perform analyses, interpret results, or produce the scientific content of the study. All scientific conclusions and intellectual contributions were made exclusively by the authors.

Publisher's Note

The publisher remains neutral regarding jurisdictional claims in published maps and institutional affiliations.

In Press

References

[1] Abualigah, L., Diabat, A., Mirjalili, S., Abd Elaziz, M., Gandomi, A. H. (2021). "The arithmetic optimization algorithm". *Computer Methods in Applied Mechanics and Engineering*, 376, Art. no. 113609. https://doi.org/10.1016/j.cma.2020.113609

[2] Akbari, M. A., Zare, M., Azizipanah-Abarghooee, R., Mirjalili, S., Deriche, M. (2022). "The cheetah optimizer: A nature-inspired metaheuristic algorithm for large-scale optimization problems". *Scientific Reports*, 12(1), Art. no. 10953. https://doi.org/10.1038/s41598-022-14338-z

[3] Ampratwum, I., Nayak, A. (2025). "Graph neural network-based internet traffic prediction in 6G networks with genetic algorithm hyperparameter optimization". In *Proc. IEEE 49th Annual Computers, Software, and Applications Conference (COMPSAC)*, pp. 100–105. https://doi.org/10.1109/COMPSAC65507.2025.00120

[4] Andi, T., Pranolo, A., Ismail, A. R., Kusuma, C. J. C. (2026). "CPSO-LSTM: Chaotic particle swarm optimization improved LSTM hyperparameters for air pollution prediction". *JOIN (Jurnal Online Informatika)*, 11(1), 126–141. https://doi.org/10.15575/join.v11i1.1689

[5] Aouedi, O., Le, V. A., Piamrat, K., Ji, Y. (2025). "Deep learning on network traffic prediction: Recent advances, analysis, and future directions". *ACM Computing Surveys*, 57(6), 1–37. https://doi.org/10.1145/3703447

[6] Ayyıldız, P., Karahan, O. (2024). "Hyperparameter optimization for a hybrid CNN-LSTM-based intrusion detection system". In *Proc. Cognitive Models and Artificial Intelligence Conference*, pp. 61–68. https://doi.org/10.36287/setsci.17.1.007

[7] BioKIDS. (2015). "Condylura cristata, star-nosed mole: Information". BioKIDS. Available: http://www.biokids.umich.edu/animals/Condylura_cristata/

[8] Cai, Z., Tan, C., Zhang, J., Zhu, L., Feng, Y. (2024). "DBSTGNN-Att: Dual branch spatial-temporal graph neural network with an attention mechanism for cellular network traffic prediction". *Applied Sciences*, 14(5), Art. no. 2173. https://doi.org/10.3390/app14052173

[9] Cao, Y., Zhang, J., Ma, A., Xu, H., Liu, J. (2026). "Marine engine cylinder exhaust temperature prediction based on PSO-optimized CNN-LSTM-attention network". *Brodogradnja*, 77(1), 1–23. https://doi.org/10.21278/brod77101

[10] Catania, K. C. (2000). "A star is born". *Natural History Magazine*, 109(6). Available: https://www.naturalhistorymag.com/picks-from-the-past/201397/a-star-is-born.

[11] Catania, K. C., Remple, F. E. (2005). "Asymptotic prey profitability drives star-nosed moles to the foraging speed limit". *Nature*, 433(7025), 519–522. https://doi.org/10.1038/nature03250

In Press

[12] Faramarzi, A., Heidarinejad, M., Stephens, B., Mirjalili, S. (2020). "Equilibrium optimizer: A novel optimization algorithm". *Knowledge-Based Systems*, 191, Art. no. 105190. https://doi.org/10.1016/j.knosys.2019.105190

[13] Farrahi Farimani, H., Bahrepour, D., Kamel Tabbakh, S. R., Ghaemi, R. (2023). "Yellow ground squirrel algorithm (YGSA): A novel metaheuristic algorithm for global optimization". *Power System Technology*, 47(4). https://doi.org/10.52783/pst.226

[14] Feldhamer, G. A., Thompson, B. C., Chapman, J. A. (Eds.). (2003). "Wild mammals of North America: Biology, management and conservation". (2nd ed.). Johns Hopkins University Press. ISBN: 9780801874161.

[15] Ghosh, A., Mangalvedhe, N., Ratasuk, R., Mondal, B., Cudak, M., Visotsky, E., …Novlan, T. D. (2012). "Heterogeneous cellular networks: From theory to practice". *IEEE Communications Magazine*, 50(6), 54–64. https://doi.org/10.1109/MCOM.2012.6211486

[16] Gong, J., Liu, Y., Li, T., Ding, J., Wang, Z., Jin, D. (2025). "STTF: A spatiotemporal transformer framework for multi-task mobile network prediction". *IEEE Transactions on Mobile Computing*, 24(5), 4072–4085. https://doi.org/10.1109/TMC.2024.3521245

[17] Guinness World Records. (2026). "Fastest Eater (Mammals)". Available: https://www.guinnessworldrecords.com/world-records/fastest-eater-(mammals).

[18] Hu, X., Liu, W., Huo, H. (2024). "An intelligent network traffic prediction method based on Butterworth filter and CNN-LSTM". *Computer Networks*, 240, Art. no. 110172. https://doi.org/10.1016/j.comnet.2023.110172

[19] Huang, Y., Li, J., Li, Y., Lin, R., Wu, J., Wang, L., Chen, R. (2024). "An improved hybrid CNN-LSTM-attention model with Kepler optimization algorithm for wind speed prediction". *Engineering Letters*, 32(10), 1957–1965. Available: http://www.engineeringletters.com/issues_v32/EL_32_4_23.pdf.

[20] Jia, H., Rao, H., Wen, C., Mirjalili, S. (2023). "Crayfish optimization algorithm". *Artificial Intelligence Review*, 56(Suppl. 2), 1919–1979. https://doi.org/10.1007/s10462-023-10567-4

[21] Jiang, W., Han, B., Habibi, M. A., Schotten, H. D. (2021). "The road towards 6G: A comprehensive survey". *IEEE Open Journal of the Communications Society*, 2, 334–366. https://doi.org/10.1109/OJCOMS.2021.3057679

In Press

[22] Jiang, W. (2022). "Cellular traffic prediction with machine learning: A survey". *Expert Systems with Applications*, 201, 117163. https://doi.org/10.1016/j.eswa.2022.117163

[23] Jiang, W., Zhang, Y., Han, H., Huang, Z., Li, Q., Mu, J. (2024). "Mobile traffic prediction in consumer applications: A multimodal deep learning approach". *IEEE Transactions on Consumer Electronics*, 70(1), 3425–3435. https://doi.org/10.1109/TCE.2024.3361037

[24] Kalita, P., Selvamuthu, D. (2023). "Stochastic modelling of sleeping strategy in 5G base station for energy efficiency". *Telecommunication Systems*, 83(2), 115–133. https://doi.org/10.1007/s11235-023-01001-9

[25] Karami, H., Anaraki, M. V., Farzin, S., Mirjalili, S. (2021). "Flow direction algorithm (FDA): A novel optimization approach for solving optimization problems". *Computers & Industrial Engineering*, 156, Art. no. 107224. https://doi.org/10.1016/j.cie.2021.107224

[26] Kaur, P., Garg, R., Kukreja, V. (2023). "Energy-efficiency schemes for base stations in 5G heterogeneous networks: A systematic literature review". *Telecommunication Systems*, 84(1), 115–151. https://doi.org/10.1007/s11235-023-01037-x

[27] Kavehmadavani, F., Nguyen, V. D., Vu, T. X., Chatzinotas, S. (2023). "Intelligent traffic steering in beyond 5G open RAN based on LSTM traffic prediction". *IEEE Transactions on Wireless Communications*, 22(11), 7727–7742, https://doi.org/10.1109/TWC.2023.3254903.

[28] Lee, K., Eo, M., Jung, E., Yoon, Y., Rhee, W. (2021). "Short-term traffic prediction with deep neural networks: A survey". *IEEE Access*, 9, 54739–54756. https://doi.org/10.1109/ACCESS.2021.3071174

[29] Lee, S., Sung, J., Shin, M. K. (2024). "Layer-wise personalized federated learning for mobile traffic prediction". *IEEE Access*, 12, 49320–49334. https://doi.org/10.1109/ACCESS.2024.3384566

[30] Li, Z., Liu, F., Yang, W., Peng, S., Zhou, J. (2021). "A survey of convolutional neural networks: Analysis, applications, and prospects". *IEEE Transactions on Neural Networks and Learning Systems*, 33(12), 6999–7019. https://doi.org/10.1109/TNNLS.2021.3084827

[31] Li, Y., Hao, M., Sun, X., Zhang, H. (2024). "Modeling super-lightweight cellular traffic prediction via multiservice and multimodal feature fusion network". *IEEE Networking Letters*, 6(1), 16–20. https://doi.org/10.1109/LNET.2023.3329744

In Press

[32] Li, S., Magli, E., Francini, G., Ghinamo, G. (2024). "Deep learning based prediction of traffic peaks in mobile networks". *Computer Networks*, 240, 110167. https://doi.org/10.1016/j.comnet.2023.110167

[33] Lin, J., Chen, Y., Zheng, H., Ding, M., Cheng, P., Hanzo, L. (2022). "A data-driven base station sleeping strategy based on traffic prediction". *IEEE Transactions on Network Science and Engineering*, 9(2), 1239–1252. https://doi.org/10.1109/TNSE.2021.3109614

[34] Liu, Z., Mao, H., Wu, C. Y., Feichtenhofer, C., Darrell, T., Xie, S. (2022). "A ConvNet for the 2020s". In *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11966–11976. https://doi.org/10.1109/CVPR52688.2022.01167

[35] Liu, J., Tang, X., Zhu, G., Cheng, X., Zhao, L., Li, H. (2024). "Multi-feature traffic prediction based on signaling information for cellular network". *IEEE Transactions on Vehicular Technology*, 73(2), 2280–2291. https://doi.org/10.1109/TVT.2023.3319351

[36] Liu, S., He, M., Wu, Z., Lu, P., Gu, W. (2024). "Spatial-temporal graph neural network traffic prediction based load balancing with reinforcement learning in cellular networks". *Information Fusion*, 103, Art. no. 102079. https://doi.org/10.1016/j.inffus.2023.102079

[37] López-Pérez, D., De Domenico, A., Piovesan, N., Xinli, G., Bao, H., Qitao, S., Debbah, M. (2022). "A survey on 5G radio access network energy efficiency: Massive MIMO, lean carrier design, sleep modes, and machine learning". *IEEE Communications Surveys & Tutorials*, 24(1), 653–697. https://doi.org/10.1109/COMST.2022.3142532

[38] Ma, X., Mu, Y., Liu, Z., Jiang, X., Zhang, J., Gao, Y. (2023). "Energy consumption optimization of 5G base stations considering variable threshold sleep mechanism". *Energy Reports*, 10, 3405–3416. https://doi.org/10.1016/j.egyr.2023.04.026

[39] Meraihi, Y., Ramdane-Cherif, A., Acheli, D., Mahseur, M. (2020). "Dragonfly algorithm: A comprehensive review and applications". *Neural Computing and Applications*, 32(21), 16625–16646. https://doi.org/10.1007/s00521-020-04866-y

[40] Minaee, S., Boykov, Y., Porikli, F., Plaza, A., Kehtarnavaz, N., Terzopoulos, D. (2021). "Image segmentation using deep learning: A survey". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(7), 3523–3542. https://doi.org/10.1109/TPAMI.2021.3059968

[41] Moghadas Gholian, S., Fiandrino, C., Vallina-Rodriguez, N., Fiore, M., Widmer, J. (2025). "DeExp: Revealing model vulnerabilities for spatio-temporal mobile traffic forecasting

In Press

with explainable AI". IEEE Transactions on Mobile Computing. https://doi.org/10.1109/TMC.2025.3531544

[42] Mousa, M. H., Algamdi, A. M., Fouad, Y., Elshewey, A. M. (2026). "CNN-MLP framework for forest burned areas prediction using PSO-WOA algorithm". *Scientific Reports*, 16. https://doi.org/10.1038/s41598-026-35836-4

[43] Nguyen, C., Hoang, T. M., Cheema, A. A. (2023). "Channel estimation using CNN-LSTM in RIS-NOMA assisted 6G network". IEEE Transactions on Machine Learning in Communications and Networking. https://doi.org/10.1109/TMLCN.2023.3278232

[44] Park, H., Yoon, S. H. (2024). "Deep reinforcement learning for base station switching scheme with federated LSTM-Based traffic predictions". *ETRI Journal*, 46(3), 379–391. https://doi.org/10.4218/etrij.2023-0065

[45] Peng, Y., Guo, Y., Hao, R., Xu, C. (2024). "Network traffic prediction with attention-based spatial-temporal graph network". *Computer Networks*, 243, Art. no. 110296. https://doi.org/10.1016/j.comnet.2024.110296

[46] Restuccia, F., Melodia, T. (2020). "Deep learning at the physical layer: System challenges and applications to 5G and beyond". *IEEE Communications Magazine*, 58(10), 58–64. https://doi.org/10.1109/MCOM.001.2000243

[47] Salahdine, F., Opadere, J., Liu, Q., Han, T., Zhang, N., Wu, S. (2021). "A survey on sleep mode techniques for ultra-dense networks in 5G and beyond". *Computer Networks*, 201, 108567. https://doi.org/10.1016/j.comnet.2021.108567

[48] Santos Escriche, E., Vassaki, S., Peters, G. (2023). "A comparative study of cellular traffic prediction mechanisms". *Wireless Networks*, 29(5), 2371–2389. https://doi.org/10.1007/s11276-023-03313-9

[49] Shabbir, A., Shirazi, M. F., Rizvi, S., Ahmad, S., Ateya, A. A. (2024). "Energy efficiency and load optimization in heterogeneous networks through dynamic sleep strategies: A constraint-based optimization approach". *Future Internet*, 16(8), Art. no. 262. https://doi.org/10.3390/fi16080262

[50] Shabbir, M., Rizvi, S., Shirazi, M. F., Alam, M. M., Su'ud, M. M. (2024). "Maximizing energy efficiency in HetNets through centralized and distributed sleep strategies under QoS constraint". *Scientific Reports*, 14, 21986. https://doi.org/10.1038/s41598-024-70714-x

In Press

[51] Shi, J., Cui, L., Gu, B., Lyu, B., Gong, S. (2023). "State transition graph-based spatial-temporal attention network for cell-level mobile traffic prediction". *Sensors*, 23(23), Art. 9308. https://doi.org/10.3390/s23239308

[52] State University of New York College of Environmental Science and Forestry (SUNY-ESF). (n.d.). "Star-Nosed Mole (Condylura cristata)". *Adirondack Ecological Center*. Available: https://www.esf.edu/aec/adks/mammals/starnosed_mole.php. Accessed: Jun. 27, 2026.

[53] Su, J., Cai, H., Sheng, Z., Liu, A. X., Baz, A. (2024). "Traffic prediction for 5G: A deep learning approach based on lightweight hybrid attention networks". *Digital Signal Processing*, 146, Art. no. 104359. https://doi.org/10.1016/j.dsp.2023.104359

[54] Sun, K., Zhang, J., Gao, X., Huang, W., Zhang, H., Leung, V. C. M. (2024). "Dynamic channel allocation scheme based on traffic prediction in dense wireless networks". *IEEE Transactions on Wireless Communications*, 23(9), 12331–12342. https://doi.org/10.1109/TWC.2024.3391426

[55] Suriyan, K., Nagarajan, R. (2024). "Particle swarm optimization in biomedical technologies: Innovations, challenges, and opportunities". In M. B. Garcia & R. P. P. Almeida (Eds.), *Emerging Technologies for Health Literacy and Medical Practice* (pp. 220–238). IGI Global Scientific Publishing. https://doi.org/10.4018/979-8-3693-1214-8.ch011

[56] Vengaimarbhan, D., Rajinigirinath, D. (2026). "Base station sleeping strategy in heterogeneous cellular Networks Using Transformer Swarm Evolutionary Adaptive Memory gate convolutional LSTM model for cellular traffic prediction". *Expert Systems with Applications*, 318, 132037. https://doi.org/10.1016/j.eswa.2026.132037

[57] Wang, Z., Hu, J., Min, G., Zhao, Z., Wang, Z., Georgalas, N. (2022). "Spatial-temporal cellular traffic prediction for 5G and beyond: A graph neural networks-based approach". *IEEE Transactions on Industrial Informatics*, 19(4), 5722–5731. https://doi.org/10.1109/TII.2022.3182768

[58] Wang, S., Yu, Z., Xu, G., Zhao, F. (2023). "Research on tool remaining life prediction method based on CNN-LSTM-PSO". *IEEE Access*, 11, 79624–79636. https://doi.org/10.1109/ACCESS.2023.3299849

[59] Wang, X., Lyu, B., Guo, C., Xu, J., Zukerman, M. (2024). "A base station sleeping strategy in heterogeneous cellular networks based on user traffic prediction". *IEEE Transactions on Green Communications and Networking*, 8(1), 134–149. https://doi.org/10.1109/TGCN.2023.3324486

In Press

[60] Wang, L., Che, L., Lam, K. Y., Liu, W., Li, F. (2024). "Mobile traffic prediction with attention-based hybrid deep learning". *Physical Communication*, 66, Art. no. 102420. https://doi.org/10.1016/j.phycom.2024.102420

[61] Wu, J., Zhang, Y., Zukerman, M., Yung, E. K. N. (2015). "Energy-efficient base-stations sleep-mode techniques in green cellular networks: A survey". *IEEE Communications Surveys & Tutorials*, 17(2), 803–826. https://doi.org/10.1109/COMST.2015.2403395

[62] Xiao, J., Cong, Y., Zhang, W., Weng, W. (2025). "A cellular traffic prediction method based on diffusion convolutional GRU and multi-head attention mechanism". *Cluster Computing*, 28(2), 125. https://doi.org/10.1007/s10586-024-04848-y

[63] Xu, Y., Gui, G., Gacanin, H., Adachi, F. (2021). "A survey on resource allocation for 5G heterogeneous networks: Current research, future trends, and challenges". *IEEE Communications Surveys & Tutorials*, 23(2), 668–695. https://doi.org/10.1109/COMST.2021.3059896

[64] Yan, B., Wang, G., Yu, J., Jin, X., Zhang, H. (2022). "Spatial-temporal Chebyshev graph neural network for traffic flow prediction in IoT-based ITS". *IEEE Internet of Things Journal*, 9(12), 9266–9279. https://doi.org/10.1109/JIOT.2021.3105446

[65] Yuan, B., Wang, J., Wu, P., Qing, X. (2021). "IoT malware classification based on lightweight convolutional neural networks". *IEEE Internet of Things Journal*, 9(5), 3770–3783. https://doi.org/10.1109/JIOT.2021.3100063

[66] Zeb, S., Rathore, M. A., Mahmood, A., Hassan, S. A., Kim, J., Gidlund, M. (2021). "Edge intelligence in softwarized 6G: Deep learning-enabled network traffic predictions". In *2021 IEEE Globecom Workshops (GC Wkshps)* (pp. 1–6). IEEE. https://doi.org/10.1109/GCWkshps52748.2021.9682131

[67] Zera, S. (2004). "Condylura Cristata (Star-Nosed Mole)". *Animal Diversity Web*, Museum of Zoology, University of Michigan. Available: https://animaldiversity.org/accounts/Condylura_cristata/. Accessed: Jun. 27, 2026.

[68] Zhang, J., Tan, C., Cai, Z., Zhu, L., Feng, Y., Liang, S. (2024). "Cellular traffic forecasting based on inverted transformer for mobile perception dual-level base station sleep control". *Ad Hoc Networks*, 161, Art. no. 103505. https://doi.org/10.1016/j.adhoc.2024.103505

[69] Zhang, R., Guo, L., Sun, J., Zhong, X., Zheng, X., Qu, L., Zhang, S., Qin, L. (2026). "CNN-LSTM model optimized by improved sparrow search algorithm for oil well production prediction". *Scientific Reports*, 16. https://doi.org/10.1038/s41598-026-43674-7

In Press

[70] Zhu, Y., Wang, S. (2023). "Traffic prediction enabled dynamic access points switching for energy saving in dense networks". *Digital Communications and Networks*, 9(4), 1023–1031. https://doi.org/10.1016/j.dcan.2022.05.017

[71] Zimmer, C. (1993). "The electric mole". *Discover Magazine*, August 1993. Available: https://www.discovermagazine.com/the-electric-mole-39973.

Authors' Bio-sketches

Hamid Rahimi is affiliated with the Department of Computer Engineering, Ma.C., Islamic Azad University, Mashhad, Iran. His research interests include artificial intelligence, machine learning, evolutionary algorithms, deep learning, optimization algorithms, and intelligent systems.

Reza Sheibani is affiliated with the Department of Computer Engineering, Ma.C., Islamic Azad University, Mashhad, Iran. His research interests include artificial intelligence, machine learning, evolutionary algorithms, computer engineering, and intelligent systems. Corresponding author. Email: reza.sheibani@iau.ac.ir

Gelareh Veisi is affiliated with the Department of Computer Engineering, Ma.C., Islamic Azad University, Mashhad, Iran. Her research interests include artificial intelligence, machine learning, evolutionary algorithms, computer engineering, and intelligent systems.

In Press