IJIE

INTERNATIONAL JOURNAL OF INTEGRATED ENGINEERING
ISSN: 2229-838X e-ISSN: 2600-7916

Vol. 18 No. 3 (2026) 340-355
https://publisher.uthm.edu.my/ojs/index.php/ijie

Multi-Algorithm Evaluation Framework for Maternal Meal Planning Based on Nutritional Needs

Diva Kurnianingtyas[1], Nathan Daud[1], Gembong Edhi Setyawan[1]*, Rakhmadhany Primananda[1]

[1] *Department of Informatics Engineering, Faculty of Computer Science, Universitas Brawijaya, 65154, INDONESIA*

*Corresponding Author: gembong@ub.ac.id
DOI: https://doi.org/10.30880/ijie.2026.18.03.023

Article Info

Received: 17 October 2025
Accepted: 12 April 2026
Available online: 30 April 2026

Keywords

Artificial intelligence, reinforcement learning, maternal issue, personalized menu, optimization

Abstract

Nutritional adequacy during pregnancy requires mothers to follow a well-balanced, personalized diet plan supporting maternal and fetal well-being. However, formulating such plans is challenging because of the combinatorial aspect of food selection and competing nutritional needs. This study offers an integrated framework of algorithms for intelligent menu planning specific to pregnant women. To measure the net optimizing value of meal recommendation programs within set nutrition limits, this study evaluated three AI programs, such as Q-Learning, SARSA, and Monte Carlo Tree Search (MCTS). Each algorithm was optimized through hyperparameter tuning to maximize a fitness function based on nutrient sufficiency, distribution, and adherence to predefined limits. Q-learning showed the fastest adaptation and performed best at elevated learning rates. In contrast, SARSA exhibited stable, more consistent behavior over extended training periods and performed best at $\alpha = 0.4$ and $\varepsilon = 0.35$. MCTS provided better performance than both reinforcement learning methods by achieving the highest fitness score through a tree-based search concerning optimal exploration weight and simulation count. These findings support the claim that each algorithm performs best in specific areas, such as Q-Learning excels at learning new information, SARSA is proficient with fixed policies, and MCTS can easily solve complex combinations of variables and meet limits. The comparative results highlight that the choice of an algorithm must be aligned with the system's goals concerning efficiency, consistency, or comprehensiveness. This study develops AI-based dietary assistance systems and demonstrates the possibilities of customized algorithms designed for effective nutrition planning in maternal healthcare.

1. Introduction

Maternal nutrition during pregnancy plays a crucial role in fetal development and maternal health by reducing the risk of low birth weight, neural tube defects, and other complications [1], [2]. However, designing well-balanced, personalized meal plans for pregnant women remains challenging due to numerous dietary restrictions, metabolic demands, and combinatorial constraints associated with multi-course meal planning [3]. Often, dietary planning is approached from a rule-based or template-based framework, resulting in a lack of flexibility and failing to account for user-specific information and preferences. These gaps have prompted the use of Artificial Intelligence (AI) technologies, such as Reinforcement Learning (RL) and heuristic search algorithms, for developing systems that provide personalized dietary recommendations [4], [5].

RL techniques, such as Q-Learning and SARSA, enable the agent to learn optimal meal recommendations through interaction with the environment and feedback in reward signals related to nutrition. While Q-Learning employs an off-policy approach that enhances sample efficiency, SARSA utilizes an on-policy approach, where learning is closely tied to actual decision-making paths. Unlike this, Monte Carlo Tree Search (MCTS) offers advantages in dealing with the combinatorial complexity of menu generation because it uses upper confidence bounds for balancing exploration and exploitation in a probabilistic, tree-based, sequential planning framework [2], [6]. Even with these advancements, little attention has been given to the comparative analysis of these AI algorithms within the framework of maternal nutrition. Most available literature focuses on the general needs of the population [7], [8] which highlights a lack of algorithmic testing for sensitive groups, such as pregnant women, whose micronutrient needs are significantly higher and require highly tailored and deeply individualized approaches [9]. In addition, pregnant women are in a vulnerable group due to physiological and public health considerations, which affect fetal development and have the potential to cause long-term effects on mothers and children [10]. This study proposes a multi-criteria algorithm evaluation approach for intelligent, constraint-aware meal planning tailored for pregnant women. This study analyzes Q-Learning, SARSA, and MCTS in terms of their ability to optimize a personalized nutrition plan using a fitness function based on nutrient completeness, dietary variety, adherence to prenatal constraints, and several other metrics. The main contributions are (1) an integrated benchmark of learning and tree search-based algorithms for nutrition planning; (2) the design and implementation of a prenatal simulation environment compliant with dietary guidelines issued by pertinent healthcare authorities; and (3) testing and evaluation of several algorithms against practical diet constraints to determine their applicability.

The remainder of this study is structured as follows: Section 2 provides an overview of the literature, which discusses the role of AI within dietary recommendation systems; Section 3 describes the methodology and provides details of the experiments conducted; Section 4 presents from an analysis of comparative performance; and Section 5 offers analysis and identifies opportunities for further research.

2. Related Works

The application of AI technology in nutrition has recently garnered significant interest, resulting in the development of various systems designed to enhance dietary recommendations. An AI-based nutrition recommendation system that combines deep generative networks and ChatGPT was implemented by [11], who created personalized weekly meal plans with high accuracy and adherence to nutritional requirements. Similarly, [12] developed NutriGen, a large language model (LLM)-based framework designed to create personalized meal plans, which improved dietary adherence. [13] proposed an LLM-powered framework for personalized food recommendation chatbots tailored to users' needs, called ChatDiet, focusing on explainability and user customization. Purrfessor, a diet health chatbot that performs visual meal analysis and gives contextual advice, was developed by [14] as a fine-tuned multimodal LLaVA. A GPT-powered AI chatbot providing tailored nutrition services for pregnant women in underserved communities was developed by [15] under the name NutritionBot.

RL has also been applied in the area of dietary planning. [16] implemented a RL based food recommendation system that utilizes the Deep Deterministic Policy Gradient (DDPG) algorithm to create personalized suggestions using users' dietary records and nutritional goals, thereby promoting better adherence to healthy eating practices. [17] conducted an extensive survey focused on recommender systems based on RL, emphasizing the usefulness of RL for the ordered, holistic, dynamic interaction between users and systems over time. [18] developed a deep RL approach for page-wise recommendations, demonstrating the usefulness of RL in recommender systems. [19] introduced a novel neural attention recommender network that accounts for user actions within the same session.

Regarding the nutrition of pregnant women, [15] developed an AI chatbot, NutritionBot, powered by GPT, which focuses on providing tailored nutrition guidance for expectant mothers. With the assistance of medical specialists, the chatbot was created to ensure the appropriateness of the dietary suggestions given. [20] centered a study on the use of ChatGPT for antenatal care, focusing on its potential to provide nutrition considerations to women with diverse health literacy levels. [21] emphasized the importance of involving specialists in AI-guided nutritional advice by discussing the human-AI collaboration approach for developing appropriate datasets for nutritional counseling. [11] reviewed personalized nutrition data technologies and relevant data collection technologies, exploring the automation of processes for tailored diet planning.

Despite the previously mentioned works, there is a notable lack of focus on applying Q-Learning, SARSA, and MCTS in the context of nutritional advice for pregnant women, which collectively constitutes a significant research gap. The purpose of this study is to address these questions by evaluating how these algorithms perform in creating personalized meal plans that meet specific dietary requirements during pregnancy.

3. Materials and Methods

To develop a comprehensive framework comprising a carefully structured nutritional dataset, multiple reinforcement learning (RL) algorithms, and a rigorous hyperparameter optimization workflow, this intelligent dietary planning methodology was formulated within a specific context. As illustrated in Fig. 1, the framework's starting point is a categorized food dataset containing core staples, plant and animal protein sources, vegetables, and supplementary foods. These categories address essential components of culturally rich prenatal diets. Regarding decision-making subsystems, the dataset serves as input, and three algorithms are implemented: Q-learning, SARSA, and MCTS. The algorithms were employed sequentially in iterative cycles to devise an optimal meal plan that adhered to specified nutritional thresholds. Each RL model underwent isolated training and optimization for its respective hyperparameters through a multifaceted nutritional fitness function centered on nutritional adequacy, rigorously balanced against all necessary constraints. Following numerous cycles of training and testing across diverse diets, the framework identified the optimal, tailored algorithmic solution for refinement in recommendations. The principal defining actions of this experiment are summarized in the following subsections.

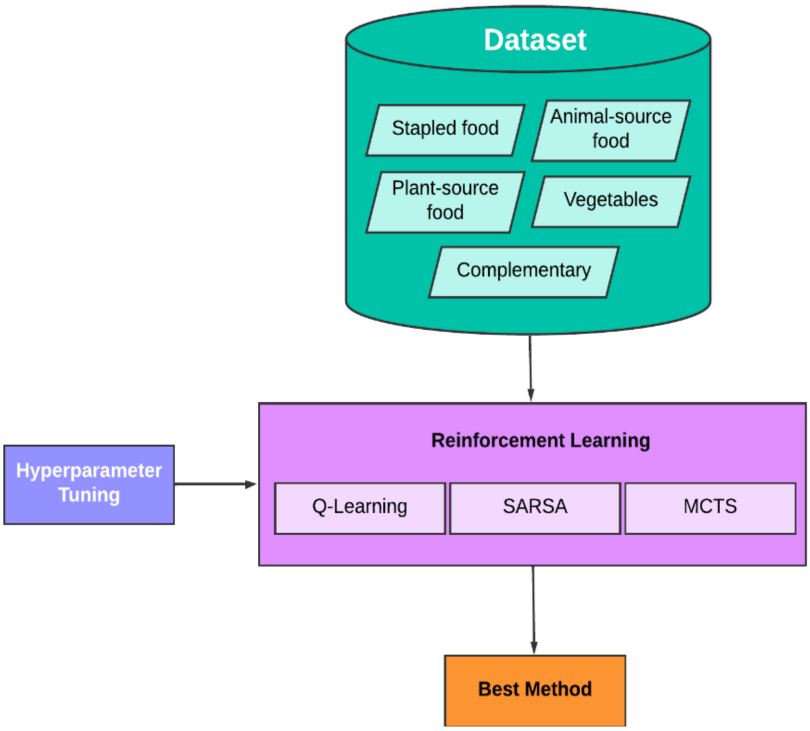


Fig. 1 *The methodology of this study*

3.1 Dataset Collection

The current study builds upon [22] article, which employed Genetic Algorithm (GA) and Particle Swarm Optimization (PSO) methodologies. [22] provides a coherent dataset that classifies food items into five overarching groups: staple foods, animal-source foods, plant-source foods, vegetables, and complementary foods. Each food item is tagged with pertinent macro and micronutrients, considering maternal nutrition, including energy, protein, fat, carbohydrates, calcium, iron, folate, and vitamin A, thereby meeting the dietary requirements of pregnant women.

3.2 Problem Formulation

In earlier studies, [22] employed PSO and GA to address the nutritional menu planning problem for pregnant women. These population-based metaheuristic approaches aim to find optimal food combinations that satisfy given constraints while ensuring maximum dietary value. Both PSO and GA effectively explore the solution space; however, their use of generational search frameworks, which lack explicit memory of past-dependent interrelations, limits their applicability to modeling time-dependent processes, such as meal planning, which requires meticulous food item selection.

To address the problem, the current study adapts the nutritional recommendation approach to a more suitable sequential decision-making problem framed within the context of RL. Unlike evolutionary algorithms, in

RL, a learning agent can interact with an environment and take feedback-driven actions, adjusting its strategy based on the long-term impact. This stepwise structure aligns with the process of constructing a meal menu, as the nutritional profile and the selection of the following food items are built cumulatively. The assignment of constructing a nutritionally balanced meal plan was approached in the context of RL as a sequential decision-making challenge for a controlling agent whose objective was to compose a menu with optimal nutritional value constrained by given limits.

This study assesses the efficiency of the three algorithms, such as Q-Learning, SARSA, and MCTS, using a simulation-based framework, where each algorithm constructs daily meal plans for pregnant women. All algorithms were pre-trained, and their performance was assessed within the same fitness evaluation framework used in prior work integrating PSO and GA [22]. This explanation facilitates consistent benchmarks in performance comparison across diverse algorithmic paradigms. Considering the actual nutrient intake, including energy, carbohydrates, protein, and fat, along with preset target requirements, the fitness function defines deviation as a quantifiable metric of deficiency. The penalty is then computed as follows in Equation (1), where C is a normalization constant set to 1000.

$$\sum Penalty = |Energy_{Need} - Calories| + |Carbohydrates_{Need} - Carbohydrates| \\ + |Protein_{Need} - Protein| + |Fat_{Need} - Fat| \quad (1)$$

Then, the overall fitness score for a complete menu is computed as Equation (2).

$$fitness = \frac{C}{\sum Penalty'}, \text{ with } C\text{=1000} \quad (2)$$

Within RL, this fitness score serves as a reward and a penalty structure within an algorithmic episode. Firstly, during an episode in which the agent has completed a full daily menu, this reward is granted after the last action contributing to the cumulative reward. The total nutritional deviation is assessed, and compensatory nutritional rewards are awarded based on the degree of nutritional deviation from a defined dietary norm. Secondly, outside of each episode, this metric focuses on the overall assessed performance of Q-Learning, SARSA, and MCTS evaluated over several iterations and scenarios. Consistent nutritional scoring across all competing algorithms ensures that the observed performance variance stems from the unique decision-making capability of each method used.

3.3 Reinforcement Learning Implementation

3.3.1 Q-Learning and SARSA

The challenge of formulating daily menus with appropriate nutrition for expectant mothers is treated in this study as a step-by-step decision-making process set within the context of a RL framework. The objective of the agent is to construct a complete menu by selecting one food item from each of the five food categories, such as staple foods, animal-source proteins, plant-based proteins, vegetables, and complementary items. Each food item is labeled with precise nutritional data, including energy (kcal), protein (g), carbohydrates (g), and fat (g). This particular form of the problem allows the agent to construct a meal plan in a sequential manner that meets the nutrition guidelines for pregnant women, especially during the second trimester.

The interaction begins with a blank slate, represented by an empty plate, and the agent must make five sequential decisions. After each decision, an agent selects one food item from an unselected category. The environment then returns to a new state that reflects the current nutritional accumulation and progress in selection, along with the corresponding reward. A state, in this case, is defined with two components: (1) the cumulative nutritional values of the selected items up to that point (for example, total energy, protein, carbohydrates, and fat) and (2) a binary vector indicating which categories have already been filled. This structure enables the algorithm to evaluate how the constructed menu meets nutritional objectives and how many potential decision spaces remain.

As an example, let us assume that the agent selects "White Rice" in the first step. This selection yields 175 kcal, 3 g protein, 40 g carbohydrates, and 0.3 g fat. Thus, the state that can be obtained after this action will look like the following Equation (3), where $State\ S1$ is the nutrition vector after selecting the first item (White Rice), consisting of cumulative nutrient values (energy, protein, carbohydrates, fat) and a category mask indicating that the “staple food” category has been filled and $State\ S2$ is the updated nutrition vector after selecting the second item (Fried Chicken), which reflects the cumulative nutrient intake from both items and a category mask showing that the “staple food” and “animal-source protein” categories have been filled.

$$State\ S1 = [175, 3.0, 40.0, 0.3, [1, 0, 0, 0, 0]] \quad (3)$$

$$State\ S2 = [350, 20.0, 40.0, 12.3, [1,1, 0, 0, 0]] \quad (4)$$

This example shows that only the staple category is filled, and the nutritional values correspond to the selected item. Now, the agent may opt for "Fried Chicken" from the animal-source category, accruing an additional 175 kcal, 17g protein, and 12g fat. The state is now updated to Equation (4). The agent persists with this procedure until it devises a full five-item menu. The agent makes a selection at each of the five decision points by taking one action, which corresponds to filling a category that has not been selected yet by choosing one item from within it. With five categories, each containing up to 15 items, the agent has an action space of roughly 70 options, of which only 15 are meaningful at each decision stage. The learning agent must find the most nutritionally optimal result from the menu within the sequence of actions taken.

In contrast to supervised learning, where correctness is verified for each sample, RL assesses correctness based on reward information. In this particular case, the reward is given as the discrepancy between the constructed menu and the preset optimal intake for the second trimester. Significant deviations from the optimal values lead to penalties as enforced by the reward function, which can use Equation (1) and (2). As a concrete example, if the final menu is shown in Table 1.

Table 1 *Example of final menu from Q-Learning and SARSA*

Menu	Total Energy (kcal)	Protein (g)	Carbohydrates (g)	Fat (g)
White rice	175	3	40	0.3
Fried Chicken	175	17	0	12
Tofu	80	8	2	5
Spinach	30	3	4	0.5
Banana	89	1	32	0.3

Upon completing an episode by choosing one food item from each of the five required categories, the agent's constructed menu is assessed for its total nutritional value. In this case, the selected items provide a total of 549 kilocalories of energy, 32 grams of protein, 69 grams of carbohydrates, and 18.1 grams of fat. These figures are then evaluated against the nutritional guidelines for pregnant women in their second trimester, which include 2250 kcal, 70 g of protein, 300 g of carbohydrates, and 70 g of fat. Calories determine the reward, and in this case, the absolute deviation from each of the four components, which sum to a target outcome of 100. Balance scoring 100 minus deviation from target nutrient components yield in Equation (5) where R denotes the reward function used within the reinforcement learning framework, which is derived from the penalty formulation in Equation (1), but reinterpreted as a reward signal. The derived outcome thus far suggests that the otherwise stated terms of the constructed reward are significant to the guidance menu, wherein nutritional balance is closely aligned with the dietary goals set, which supports the learning process through further exploration in future learning episodes. The agent, with the help of RL techniques, selects preferred options for menu compilation that adhere more closely to prescribed allowances, thereby refining the quality of its recommendations.

$$\begin{aligned} R &= 100 - (|2250 - 549| + |70 - 32| + |300 - 69| + |70 - 18.1|) \\ &= 100 - (1701 + 38 + 231 + 51.9) \\ &= -1921.9 \end{aligned} \quad (5)$$

Both Q-learning and SARSA improve their action-value evaluations based on the same reward signal, but they differ in how they predict future rewards. In Q-Learning's off-policy approach, the reward for a given state-action pair is updated based on the maximum anticipated reward for the subsequent state, irrespective of the action in Equation (6). SARSA, on the other hand, relies on an on-policy update and adjusts the value based on what action is taken to the next step in Equation (7). This difference makes Q-learning more exploratory and potentially speeds up convergence to an optimal policy, whereas SARSA results in more conservative and stable learning trajectories. This characteristic is critical for dietary systems where long-term consistency is essential. Although the action space is combinatorial, state abstraction and discretization constrain the total number of state-action pairs to a computable level. Thus, Q-Learning and SARSA can be effectively and transparently used to devise a dynamically adapted, personalized nutritional recommendation system designed for the specific needs of pregnancy.

$$Q(s, a) \leftarrow Q(s, a) + \alpha[r + \gamma \cdot \max_{a'} Q(s', a') - Q(s, a)] \quad (6)$$

$$Q(s,a) \leftarrow Q(s,a) + \alpha[r + \gamma \cdot Q(s',a') - Q(s,a)] \quad (7)$$

where:
$Q(s,a)$ = the action-value function representing the expected cumulative reward for taking action a in state s,
α = the learning rate that controls how much new information overrides old estimates ($0 < \alpha \leq 1$),
r = the immediate reward received after taking action aa in state s,
γ = the discount factor ($0 \leq \gamma \leq 1$) that determines the importance of future rewards,
$Q(s',a')$= the estimated action-value function for the next state s' and action a', used to update the current state-action value.

3.3.2 MCTS

This study examines the application of MCTS in formulating daily dietary menus for pregnant women within a decision tree framework for food selection. Unlike Q-Learning and SARSA, which gradually update action values in real-time, MCTS operates differently; it simulates the launch of a full course menu and refines the decision route using statistical backpropagation. The agent's task is to compose a five-item menu that includes one item from each of the designated five nutritional categories: staple foods, animal proteins, vegetable proteins, vegetables, and complementary foods. The simulation is static, and the menu structure is fixed, as a selection must consist of five items divided into the listed categories. At any stage within the simulation, the state is defined as the selected food items, a category mask denoting which nutrition categories have been selected, and the running total of the specified aggregate nutritional value. Let us assume that the agent partitions the selection into two steps with "White Rice" and "Fried Chicken", the state would comprise these two items, a category mask of [1, 1, 0, 0, 0], and nutrient totals superficially like 350 kcal energy, protein 20g, carbohydrates 40g, and 12.3g fat. At this stage, an action involves selecting a menu food item from a category that is not yet complete, in our case, vegetable protein. Possible actions include "Tofu", "Tempeh", or "Soy Milk", each of which adds another nutrient and branches of the decision tree. This process continues until the agent completes the episode by selecting from each of the five categories. At that moment, the reward is earned by evaluating the metabolic dietary allowances calculation from the menu against the pregnancy-specific dietary targets, as explained in Equation (5).

These very strongly negative rewards highlight insufficient progression toward the distinct nutritional goals one is intended to achieve, thus causing the agent to decline that course of action. During backpropagation, rewards are distributed up the tree as updates to the visit counts and average values of the relevant nodes. This process enables MCTS to direct many simulations towards more promising action sequences that lead to nutritionally balanced menus. The agent independently discovers optimal combinations over several iterations using trial-and-error learning, which does not involve explicit strategy incidence mapping, population, or parameter adjustment, making the MCTS especially well-suited for problems exhibiting high combinatorial complexity and sparse rewards.

3.4 Hyperparameter tuning configuration

3.4.1 Q-Learning

The Q-Learning algorithm implementation requires systematic optimization of four critical hyperparameters: the learning rate (α), the discount factor (γ), the exploration rate (ε), and the number of training episodes. The tuning methodology employs a comprehensive grid search approach to explore various combinations of these parameters and identify the optimal configuration for menu recommendation tasks. The hyperparameter tuning process begins by using default values for each parameter, specifically $\alpha = 0.01$, $\gamma = 0.6$, $\varepsilon = 0.01$, and episodes = 1000. The tuning is conducted sequentially for each hyperparameter, starting with parameter α, followed by γ, ε, episodes. When the optimal value for one hyperparameter is found, that value is used as a fixed setting for tuning the next hyperparameter. In contrast, hyperparameters that have not yet been tuned continue to use their default values. The details of the hyperparameter tuning configuration are presented in Table 2.

Table 2 *Q-Learning hyperparameter tuning configuration*

Hyperparameter	Value
α	[0.01, 0.03, 0.05, 0.07, 0.1, 0.15, 0.2, 0.3, 0.4, 0.5]
γ	[0.6, 0.65, 0.7, 0.75, 0.8, 0.85, 0.9, 0.92, 0.95, 0.99]
ε	[0.01, 0.03, 0.05, 0.07, 0.1, 0.15, 0.2, 0.25, 0.3, 0.35]
Episodes	[1000, 2000, 3000, 4000, 5000, 6000, 7000, 8000, 9000, 10000]

For the α parameter, multiple values are systematically tested to determine the optimal balance between learning speed and stability. Lower α values provide more stable learning but slower convergence, while higher values accelerate learning but may cause instability. The tuning process evaluates different α ranges to determine the optimal setting for this specific application. The γ tuning focuses on balancing immediate nutritional rewards versus long-term dietary benefits, testing various values to optimize the algorithm's consideration of future nutritional outcomes in menu planning. The ε optimization involves testing different values to achieve the optimal balance between exploration and exploitation. This parameter ensures the algorithm explores diverse menu combinations while exploiting known practical recommendations. The episode parameter tuning determines the sufficient training duration for convergence, testing various episode counts to identify when the algorithm reaches stable performance without unnecessary computational overhead.

3.4.2 SARSA

The SARSA implementation follows a similar hyperparameter tuning approach but considers the unique characteristics of on-policy learning. The hyperparameter tuning process for SARSA begins with default values for each parameter, specifically α = 0.01, γ = 0.6, ε = 0.01, and episodes = 1000. Following the same sequential optimization methodology as Q-Learning, the tuning is conducted systematically for each hyperparameter in order: α, followed by γ, ε, and finally, the number of episodes. Once the optimal value for one hyperparameter is identified, it is fixed and used to tune subsequent parameters. In contrast, hyperparameters that have not yet been optimized continue to use their default values. The four primary parameters, such as α, γ, ε, and episodes, are systematically optimized, with particular attention to how the on-policy nature of the algorithm affects the learning process. The details of the hyperparameter tuning configuration are presented in Table 3.

The α parameter tuning for SARSA requires careful consideration of the fact that the algorithm learns from the actual policy being followed rather than the optimal policy. Given the on-policy updates, this statement necessitates testing learning rates that provide stable convergence. The γ optimization focuses on ensuring appropriate weighting of future nutritional benefits while considering that SARSA's conservative approach may require different discount factor values compared to Q-Learning. The ε tuning in SARSA is particularly critical since the exploration rate directly affects both the learning process and the policy being learned. The methodology tests various ε values and decay strategies to find the optimal balance between exploration and exploitation, specifically for the on-policy learning context. The episode parameter optimization ensures sufficient training time for the more conservative SARSA approach to converge effectively.

Table 3 *SARSA hyperparameter tuning configuration*

Hyperparameter	Value
α	[0.01, 0.03, 0.05, 0.07, 0.1, 0.15, 0.2, 0.3, 0.4, 0.5]
γ	[0.6, 0.65, 0.7, 0.75, 0.8, 0.85, 0.9, 0.92, 0.95, 0.99]
ε	[0.01, 0.03, 0.05, 0.07, 0.1, 0.15, 0.2, 0.25, 0.3, 0.35]
Episodes	[1000, 2000, 3000, 4000, 5000, 6000, 7000, 8000, 9000, 10000]

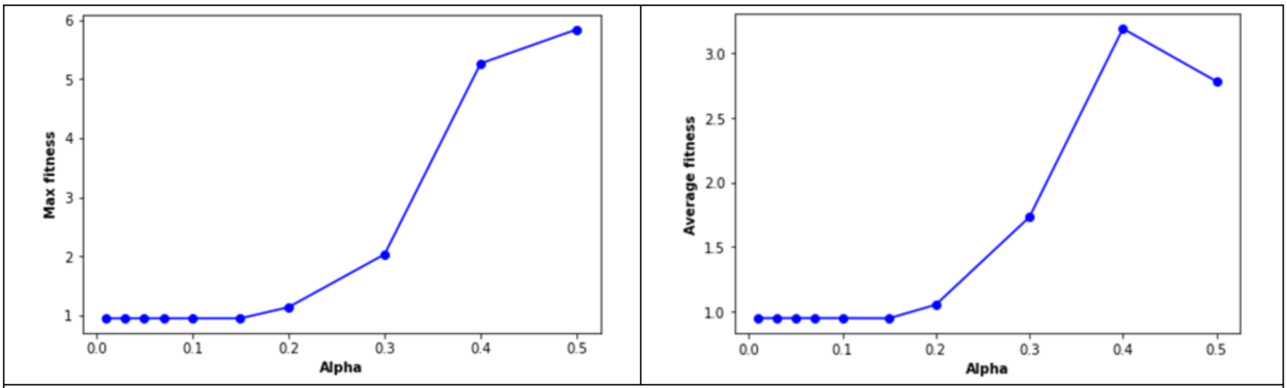
3.4.3 MCTS

The MCTS implementation requires a fundamentally different tuning approach compared to the value-based RL methods. The hyperparameter tuning process for MCTS begins with default values for each parameter, specifically exploration_weights = 0.5 and num_simulations = 10. The tuning is conducted sequentially, starting with the exploration_weights parameter and then the num_simulations parameter. When the optimal value for one hyperparameter is identified, it is fixed and used to tune the subsequent parameter. The primary hyperparameters for optimization are exploration weights and the number of simulations per decision. This approach focuses on optimizing the MCTS components rather than traditional RL parameters. The details of the hyperparameter tuning configuration are shown in Table 4.

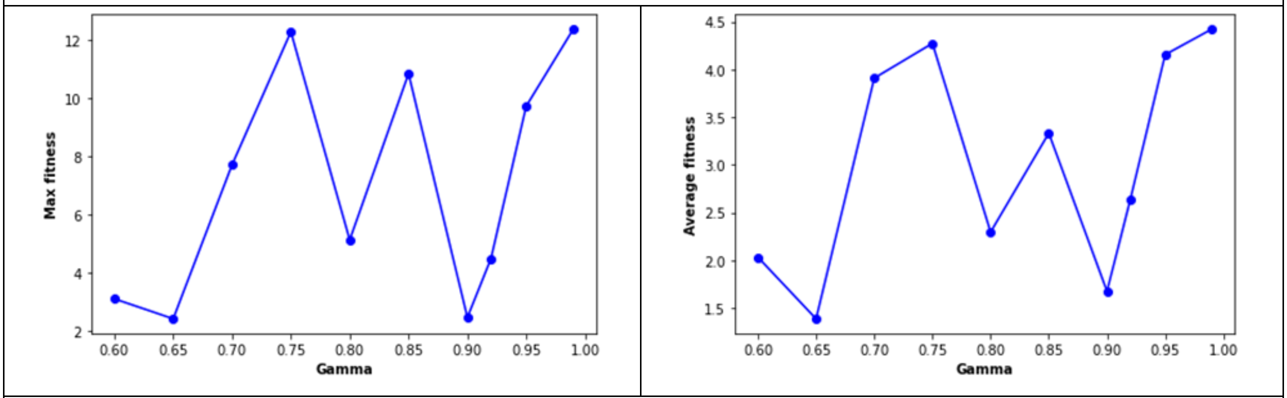
Table 4 *MCTS hyperparameter tuning configuration*

Hyperparameter	Value
exploration_weights	[0.5, 1.0, 1.5, 2.0, 2.5, 3.0, 3.5, 4.0, 4.5, 5.0]
num_simulations	[10, 25, 50, 75, 100, 150, 200, 250, 300, 350]

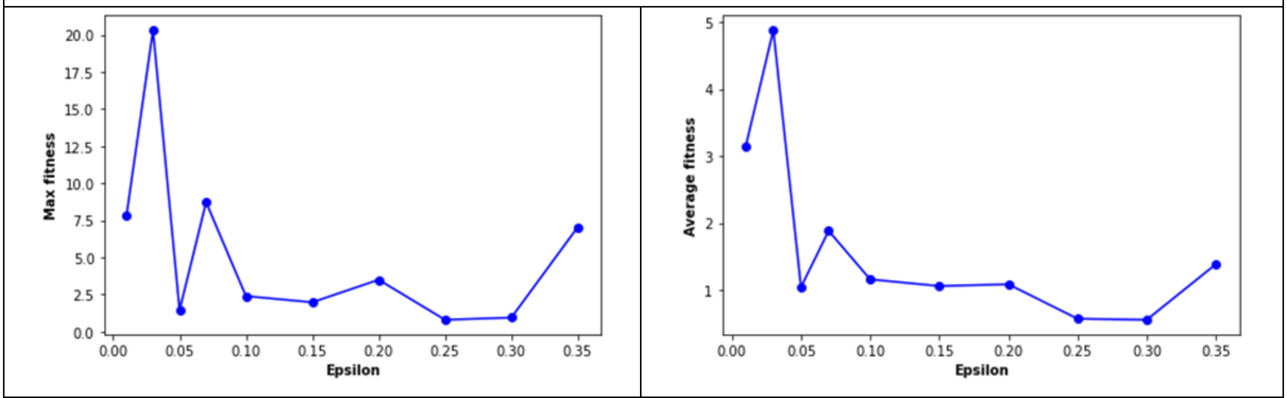
The exploration weights parameter tuning determines the balance between exploring new menu combinations and exploiting known good recommendations within the tree search framework. The methodology tests various exploration weight values to ensure comprehensive coverage of the recommendation menu while maintaining computational efficiency. This parameter is crucial for the Upper Confidence Bound (UCB) formula used in MCTS to select promising nodes for expansion. The number of simulation parameter optimizations involves finding the optimal balance between recommendation quality and computational cost. The tuning process systematically varies the simulation count to determine the minimum number required for reliable performance while considering computational constraints. More simulations generally lead to better decision quality, but they require proportionally more computational resources, making this optimization critical for practical implementation.



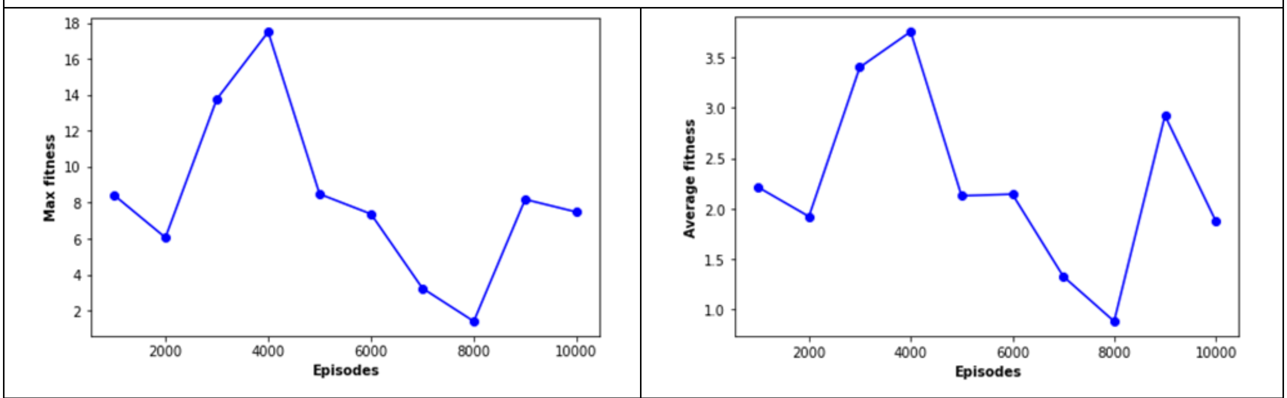
(a) α parameter



(b) γ parameter

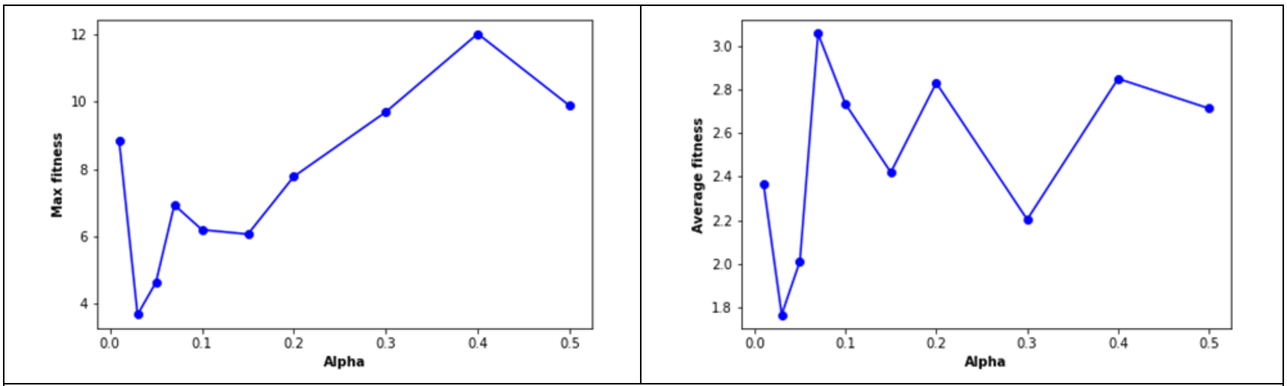


(c) ε parameter

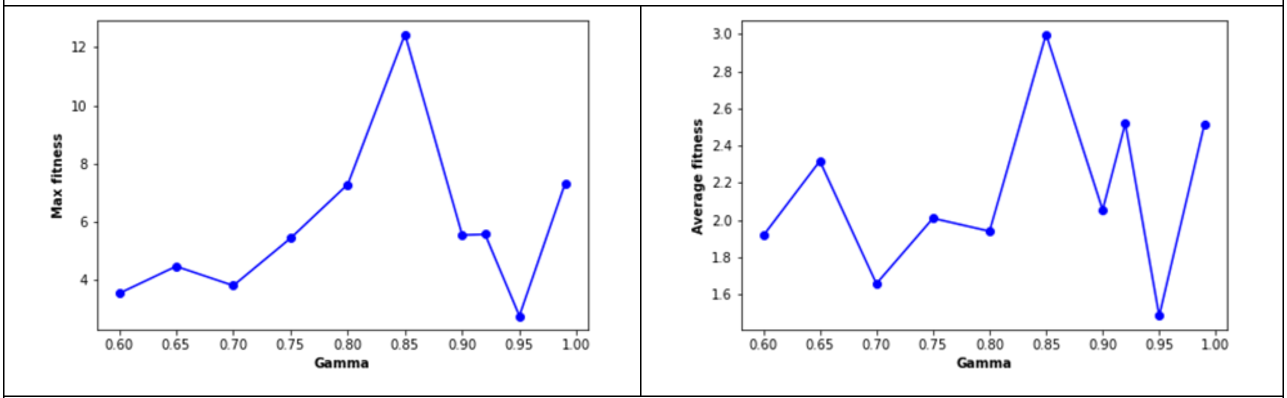


(d) episode parameter

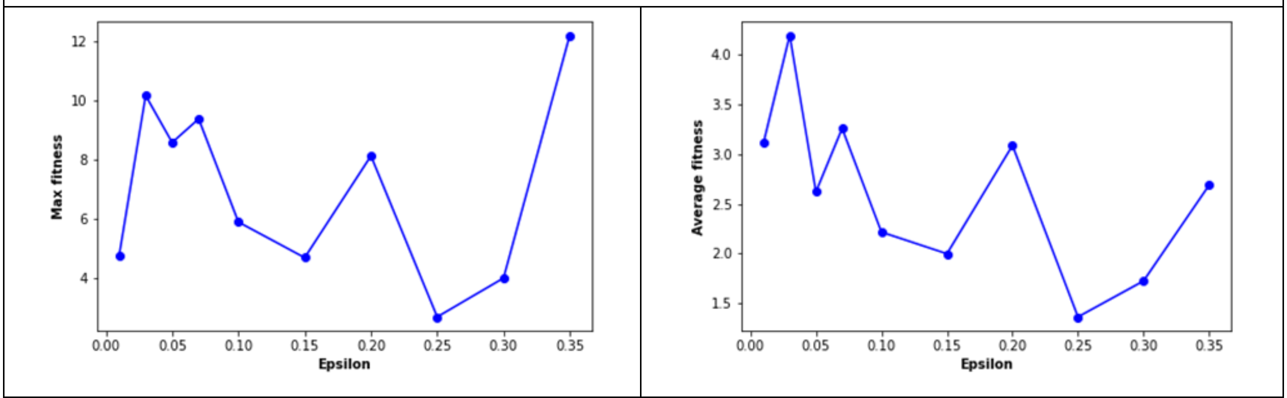
Fig. 2 *Q-Learning hyperparameter optimization results*



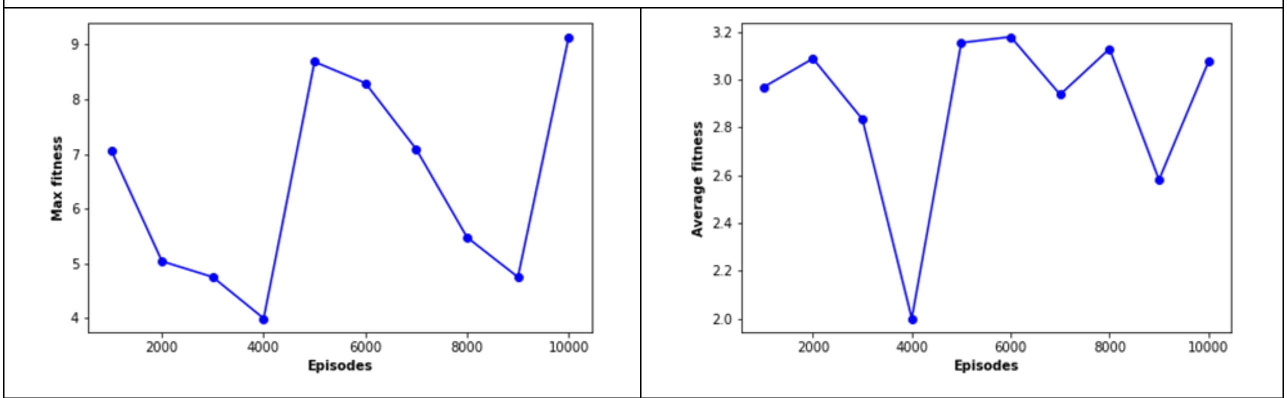
(a) α parameter



(b) γ parameter



(c) ε parameter



(d) episode parameter

Fig. 3 *SARSA hyperparameter optimization results*

3.5 Evaluation

The configuration of hyperparameters for Q-Learning and SARSA included adjusting the α alongside the γ, the ε, and the number of episodes. For MCTS, this study adjusted the number of simulations conducted for each decision alongside the exploration_weight used in the UCB formula. Each tested algorithm was explored under three experimental scenarios, such as (1) A baseline menu with standard pregnancy nutrient targets, (2) A trimester-specific focus with increased emphasis on iron and folate during one of the trimesters, and (3) a calorie-restricted scenario designed to simulate controlled gestational weight. For the evaluation of each configuration, this study conducted a maximum of 10 iterations to accommodate variability. The results for each configuration were then combined at a 95% confidence level. The metrics recorded are as follows: maximum fitness, average fitness, convergence stability, training time, and the constraint satisfaction rate, which is defined as the ratio of the total number of menus generated to the percentage of menus that fully adhered to all stringent nutritional limits

4. Results

4.1 Hyperparameter Tuning Result

The systematic hyperparameter tuning process conducted for all three RL algorithms yielded significant insights into their optimal configurations for nutritional menu recommendation systems. The comprehensive grid search methodology employed across Q-Learning, SARSA, and Monte Carlo Tree Search (MCTS) algorithms reveals varying sensitivity patterns to different parameter configurations, reflecting the unique characteristics of each learning approach in the context of dietary recommendations for pregnant women.

4.1.1 Q-Learning Performance Analysis

The Q-Learning algorithm demonstrated robust performance characteristics through the hyperparameter optimization process, as comprehensively illustrated in Fig. 2. The α tuning revealed optimal convergence patterns that balance learning stability with adaptation speed, which is crucial for handling pregnant women's complex nutritional requirement space. Fig. 2(a) demonstrates how the α parameter significantly influences the algorithm's ability to update Q-values effectively while maintaining stability in the learning process. The γ optimization results, shown in Fig. 2(b), indicate the algorithm's sensitivity to future reward consideration, which is particularly relevant in nutritional planning, where long-term dietary benefits must be balanced against immediate nutritional needs. Fig. 2(c) presents the ε tuning, demonstrating the importance of maintaining adequate exploration in the menu recommendation space to avoid premature convergence to suboptimal dietary combinations. The episode parameter optimization, depicted in Fig. 2(d), reveals the training duration required for stable convergence, providing insights into the computational efficiency of the Q-Learning approach for this specific application domain.

Table 5 *Optimal hyperparameter for Q-Learning*

Hyperparameter	Value
α	0.4
γ	1
ε	0.03
episodes	4000

The hyperparameter sensitivity analysis revealed distinct optimization patterns across all tested parameters. The alpha parameter demonstrated a clear optimal range between 0.4 and 0.5, where learning stability was maintained without sacrificing convergence speed. The γ parameter optimization revealed that a value of 1 provided the best balance between immediate and future nutritional rewards. ε- decay strategies proved crucial, notably by limiting ε to 0.35, which prevented excessive random exploration that could generate impractical meal combinations. With initial exploration rates of 0 gradually increasing to 0.35, this approach yielded optimal exploration-exploitation trade-offs. The episode analysis confirmed that 4000 episodes were sufficient for stable convergence while maintaining computational efficiency. The random initialization process means that higher episode counts do not necessarily guarantee better performance, as the quality of initial solutions significantly influences the learning trajectory. Episodes beyond approximately 8000-10000 may exhibit decreased performance due to overfitting or convergence to local optima determined by the initial random state. Table 5 summarizes the optimal hyperparameter configuration that maximizes the fitness function while ensuring practical computational requirements. The selected parameters represent the best compromise between learning performance, computational efficiency, and practical implementation considerations for nutritional recommendation systems.

4.1.2 SARSA Performance Analysis

The SARSA algorithm's on-policy learning characteristics yielded distinct hyperparameter sensitivity patterns compared to Q-Learning, as illustrated in Fig. 3. The conservative nature of SARSA's learning approach, which learns from the actual policy being followed rather than the optimal policy, necessitated different parameter configurations to achieve optimal performance in menu recommendation tasks. Fig. 3(a) illustrates the learning rate optimization for SARSA, revealing the need for more conservative α values due to the algorithm's dependency on the current policy for learning updates. This characteristic proved beneficial for nutritional recommendation systems where stability and consistency in dietary suggestions are paramount. Fig. 3(b) shows the discount factor tuning, highlighting SARSA's unique approach to balancing immediate nutritional rewards with long-term dietary planning considerations. Fig.3(c) and 3(d) illustrate the exploration rate and episode optimization, respectively, demonstrating SARSA's sensitivity to ε values, as the exploration strategy directly impacts both the learning process and the policy being learned. This dual impact makes ε tuning critical for achieving optimal performance in the nutritional recommendation domain.

Table 6 *Optimal hyperparameter for SARSA*

Hyperparameter	Value
α	0.07
γ	0.85
ε	0.03
episodes	6000

The analysis revealed that SARSA requires more conservative parameter settings than Q-Learning. The optimal α values were consistently lower (0.07), reflecting the algorithm's need for stable, gradual learning updates. The γ parameter exhibited similar patterns to Q-Learning but preferred slightly higher values (0.85), highlighting the importance of future rewards in the on-policy learning context. The ε parameter required careful tuning to balance the algorithm's exploration needs with its conservative learning approach, with optimal values 0.03, exhibiting a gradual decay. The ε parameter range was limited to 0.35 to prevent excessive random exploration that could disrupt SARSA's on-policy learning stability and generate impractical meal combinations. As shown in the optimization curves, the episode parameter results confirm the algorithm's requirement for extended training periods to achieve stable convergence, reflecting the more conservative learning approach inherent to SARSA. Table 6 presents the optimal configuration parameters derived from this comprehensive analysis.

4.1.3 MCTS Performance Analysis

The MCTS algorithm's approach to the menu recommendation challenge prompted a different hyperparameter optimization strategy centered around focus weights and simulation counts instead of RL parameters, as shown in Fig. 4. The tree search approach has proven to be very useful for solving the combinatorial problem arising from menu planning while providing a reasonable level of efficiency in the computation. Fig. 4(a) illustrates how exploration weights are optimized, demonstrating the sensitivity of the algorithm to the trade-off between exploring additional menu combinations and exploiting practical recommendations already known to be effective within the tree search framework. The balance concerning the exploration of new or previously recommended menus is notable, given the dependence of the Upper Confidence Bound (UCB) formula on this parameter, as it relates to the coverage of the nutritional recommendation space within the practical limits of computational resources. Fig. 4(b) shows the simulation count optimization and demonstrates the trade-off between the widely accepted quality of recommendations and the computational cost of MCTS techniques. These results suggest a minimum simulation threshold for dependable performance in the context of practical constraints implementing nutritional recommendation systems.

Table 7 *Optimal hyperparameter for MCTS*

Hyperparameter	Value
exploration_weights	5
num_simulations	200

Identifying distinct characteristics that differentiate MCTS optimization from value-based RL techniques revealed a form of value specialization within MCTS. The exploration weight parameter is optimized at 5. At this value, the UCB formula balanced the exploitation of well-performing menu recommendations and already-proven combos with new menu exploration perfectly. Higher values of exploration weights resulted in excessive overhead from exploration. In comparison, lower values led to submenu convergence to combinations of suboptimal parts that were too far off to be optimal. The simulation count analysis provided further insights, showing a clear tradeoff between achievement accuracy and decision time. These 200 simulations per decision yielded the highest accuracy within the given time frame". Table 7 consolidates the optimal settings found, along with the restrictions on computational and practical recommendation efficiency that enable real-world workability.

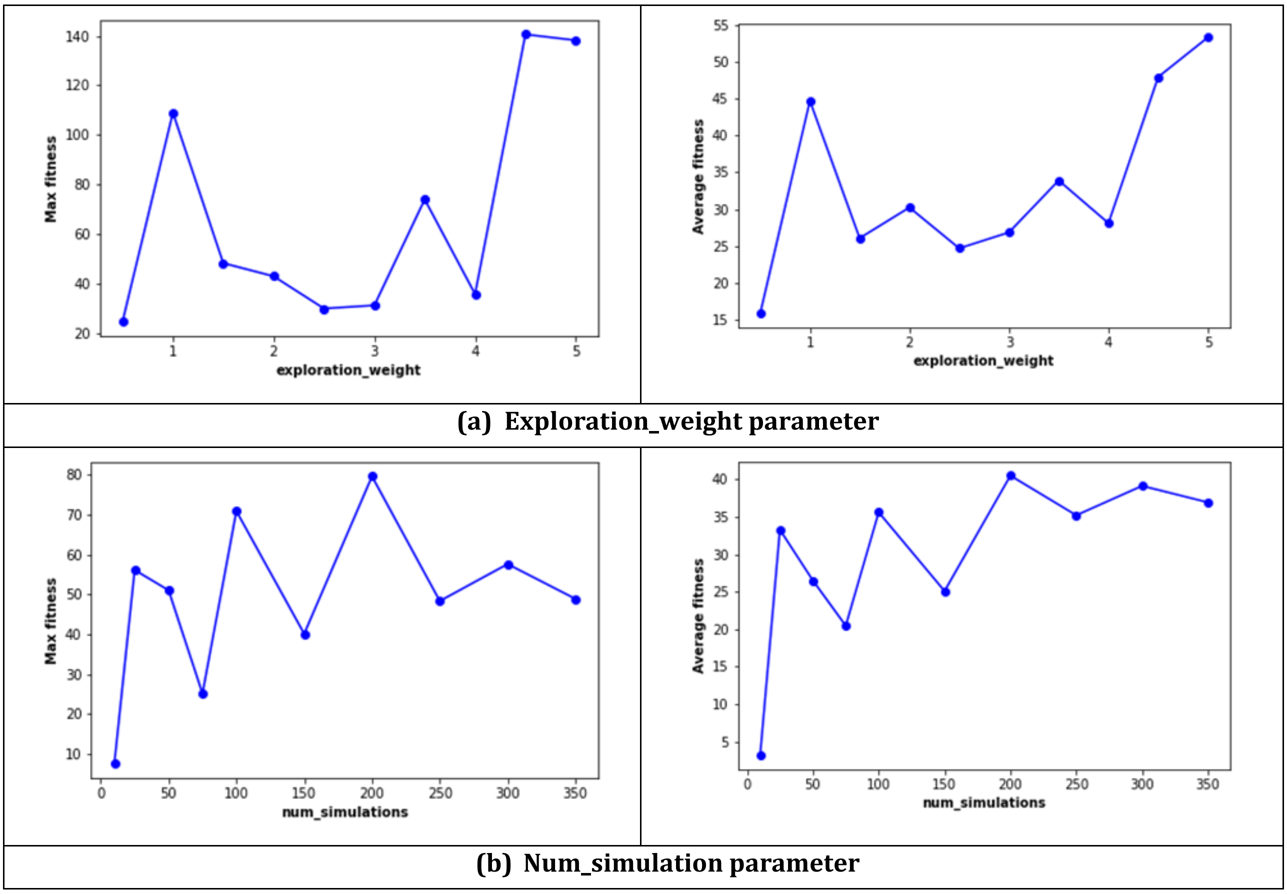


Fig. 4 *SARSA hyperparameter optimization results*

4.2 Comparative Performance Analysis

The application-based comparative evaluation of the three algorithms for nutritional menu recommendation systems demonstrates unique merits and features, as systematically captured in Table 8. Q-Learning's off-policy exploration biases during the exploratory phase yield better sample efficiency across the nutritional landscape compared to SARSA's on-policy strategy, which tends to yield better stability and consistency in alignment with real-life recommendation adherence. MCTS's tree search approach exhibits notable strengths in dealing with the combinatorial complexity of menu planning by offering a structured yet computationally feasible exploration of dish combinations. The algorithm's achievement in balancing exploration and exploitation through the UCB formula uniquely guarantees comprehensive coverage of minimum nutritional requirements without failure. The analysis of competitiveness reveals that all the focused algorithms possess distinct strengths in the nutritional recommendation problem. The best Q-fit results in less than optimal time suggest that Q-Learning can quickly learn and adjust to new information. It was observed that SARSA maintained the best reliability across a range of scenarios, making it ideal for environments that require enduring, consistent, and highly accurate recommendations. The ability demonstrated by MCTS to manage complicated and interdependent constraints on the menu was remarkable, especially in terms of advanced nutritional planning.

Table 8 *Reinforcement learning comparison results*

Algorithms	Fitness
Q-Learning	13.1101
SARSA	9.8112
MCTS	98.5658

5. Discussion

The acquired data suggests notable variations in patterns of behavior and the effectiveness of each algorithm's performance. While Q-Learning achieves moderate levels of fitness, it is susceptible to hyperparameters, particularly the exploration rate, which can sometimes lead to convergence issues. Counteracting this weakness, though, is the markedly accelerated learning that the algorithm offers due to off-policy benefits, which render it useful when a quick adjustment is desired, even when the computational expense is moderate. This finding is consistent with a study by [23], which emphasizes the efficiency of Q-Learning in achieving optimal solutions, especially in dynamic environments, despite its sensitivity to exploration parameters.

On the other hand, SARSA yielded lower fitness values but improved reliability. These results stemmed from the on-policy structure of SARSA, as it learned from the action paths taken and reinforced learning, resulting in reliable output, albeit with lower fitness values. Previous studies by [24] also show that SARSA tends to choose safer and more consistent paths, avoiding exploration risks that could result in greater penalties. SARSA is more stable in environments that prioritize consistency of action, although it is not always optimal in terms of performance. For use-case solutions, SARSA is attractive in scenarios where repeatability and reliability take precedence over aggressively optimized performance. In this study, MCTS appeared to be the most optimal algorithm as it provided the highest fitness values with smoother trends over varying hyperparameter values. Its tree-based planning approach provided sufficient exploration and exploitation, with adequate fulfillment of multi-objective constraints for meal planning using UCB. These results significantly highlight the applicability of MCTS in complex and combinatorial optimization issues in nutrition planning, especially where the computational burden poses no limit. This finding reinforces [25] results, which demonstrate the superiority of MCTS for simulation-based planning through gradual exploration of decision trees. MCTS is considered a powerful and versatile tool in the domain of multi-objective planning and decision-making.

New revelations emerge when juxtaposed with previous work utilizing GA and PSO. The merit PSO has received for achieving average levels of fitness is noteworthy. Regardless, the concern remains. Lower execution time does not equate to adaptive learning, a requirement in RL frameworks. In contrast, GA's greater computational demand led to a lower fitness score. Firm reliance on population-based optimization frameworks is what makes both GA and PSO convergent to these solution paradigms. These frameworks treated menus as static combination sets, voiding flexibility to adapt during the iterative meal planning process. These findings are consistent with studies by [26] for PSO and [27] for GA, which show that although these approaches are efficient in exploring solution space, they are less responsive to environmental changes.

Evaluating the decision-making process in a stepwise fashion and recalculating estimates dynamically, MCTS and other RL frameworks incorporate timelessness as they adjust strategies based on the comprehensive impact of nutrients. The evolution from generational to sequential learning frameworks represents a significant leap in systems designed for user-specific recommendations, particularly within maternal healthcare, where nutritional needs change during the trimesters of pregnancy. This finding is reinforced by [28], [29] who emphasize the superiority of MCTS in simulation-based and multi-objective decision making for various purposes and scenarios depending on the application domain. Moving away from PSO and GA towards RL not only augments result and refines performance but also enhances interpretability, real-time responsiveness, seamless integration, and system flexibility. For these reasons, MCTS and RL frameworks are the most advanced intelligent systems for nutrition planning, where data and responsiveness are critical.

6. Conclusion

This study examines the application of three different RL algorithms, Q-Learning, SARSA, and MCTS, to create an optimized nutritional menu for pregnant women. It builds off existing data and prior research, which addressed this problem using PSO and GA. It reformulates the problem as a subprocess of a hierarchical sequential decision-making framework enabling adaptive and dynamic recommendations.

The results obtained show that MCTS exhibits a decisive advantage over SARSA and Q-Learning in all configurations tested, achieving a maximum nutritional fitness score of 98.56. Q-Learning outperformed expectations with the calibrated hyperparameters for learning; however, it resulted in no consistency. SARSA yielded more consistent results but an overall lower performance. Unlike other approaches, MCTS reached PSO comparable values while employing a strategic planning approach, making the method more flexible and controllable. The uniform application of the same fitness function enabled a relative comparison and evaluation of the results. These results illustrate the capabilities of RL models in capturing temporal dependencies as well as the cumulative trade-off of nutrition over time and across meals. It shows a shift in methodology from population-based optimization toward incremental adaptations based on learned intelligence. Further studies consider integrating vitamins such as iron and folate, or sodium, into the trimester-based dynamic objectives or incorporating user-specific dietary limitations and preferences into the reward system. Additional personalization and generalizability could be attained through deep RL, as well as expanding the framework to multi-day meal planning. This study highlights the potential of RL and MCTS, particularly as a viable approach to developing nutrition recommendation systems within the context of maternal health and other domains where AI plays a significant role.

Acknowledgement

This work was supported by Faculty of Computer Science, Universitas Brawijaya under *Hibah Doktor Non Lektor Kepala Tahun* 2024 Grand number 02251/UB10.F1501/B/PT.01.05.1/2024.

Conflict of Interest

Authors declare that there is no conflict of interest regarding the publication of the paper.

Author Contribution

The authors confirm contribution to the paper as follows: ***study conception, design, analysis, interpretation, and draft manuscript preparation:*** *Diva Kurnianingtyas;* ***data collection, software, results, and draft manuscript preparation:*** *Nathan Daud;* ***draft manuscript preparation:*** *Gembong Edhi Setyawan, Rakhmadhany Primananda*

References

[1] R. Yasin, M. Azhar, Z. Allahuddin, J. K. Das, and Z. A. Bhutta, "Antenatal care strategies to improve perinatal and newborn outcomes," *Neonatology*, vol. 122, no. Suppl. 1, pp. 13–31, 2025.

[2] D. Tsolakidis, L. P. Gymnopoulos, and K. Dimitropoulos, "Artificial Intelligence and Machine Learning Technologies for Personalized Nutrition: A Review," in *Informatics*, MDPI, 2024, p. 62.

[3] U. M. Leonard and M. E. Kiely, "Can micronutrient requirements be met by diets from sustainable sources: outcomes of dietary modelling studies using diet optimization," *Ann Med*, vol. 56, no. 1, p. 2389295, 2024.

[4] S.-T. Cheng, Y.-J. Lyu, and C. Teng, "Image-Based Nutritional Advisory System: Employing Multimodal Deep Learning for Food Classification and Nutritional Analysis," *Applied Sciences*, vol. 15, no. 9, p. 4911, 2025.

[5] G. Prasad, M. Padhiary, A. Hoque, and K. Kumar, "AI-Driven Personalized Nutrition Apps and Platforms for Enhanced Diet and Wellness," in *Food in the Metaverse and Web 3.0 Era: Intersecting Food, Technology, and Culture*, IGI Global Scientific Publishing, 2025, pp. 125–158.

[6] A. G. Barto, "Reinforcement learning: An introduction. by richard's sutton," *SIAM Rev*, vol. 6, no. 2, p. 423, 2021.

[7] H. Ko, S. Lee, Y. Park, and A. Choi, "A survey of recommendation systems: recommendation models, techniques, and application fields," *Electronics (Basel)*, vol. 11, no. 1, p. 141, 2022.

[8] A. Mate, C. Reyes-Goya, Á. Santana-Garrido, and C. M. Vázquez, "Lifestyle, maternal nutrition and healthy pregnancy," *Curr Vasc Pharmacol*, vol. 19, no. 2, pp. 132–140, 2021.

[9] M. Puwanant, S. Boonrusmee, S. Jaruratanasirikul, K. Chimrung, and H. Sriplung, "Dietary diversity and micronutrient adequacy among women of reproductive age: a cross-sectional study in Southern Thailand," *BMC Nutr*, vol. 8, no. 1, p. 127, 2022.

[10] D. Kurnianingtyas and L. Muflikhah, "Application of Artificial Intelligence for Maternal and Child Disorders in Indonesia: A Review," in *Asia Simulation Conference*, Springer, 2024, pp. 289–306.

[11] I. Papastratis, D. Konstantinidis, P. Daras, and K. Dimitropoulos, "AI nutrition recommendation using a deep generative model and ChatGPT," *Sci Rep*, vol. 14, no. 1, p. 14620, 2024.

[12] S. Khamesian, A. Arefeen, S. M. Carpenter, and H. Ghasemzadeh, "NutriGen: Personalized Meal Plan Generator Leveraging Large Language Models to Enhance Dietary and Nutritional Adherence," *arXiv preprint arXiv:2502.20601*, 2025.

[13] Z. Yang *et al.*, "ChatDiet: Empowering personalized nutrition-oriented food recommender chatbots through an LLM-augmented framework," *Smart Health*, vol. 32, p. 100465, 2024.

[14] L. Lu, Y. Deng, C. Tian, S. Yang, and D. Shah, "Purrfessor: A Fine-tuned Multimodal LLaVA Diet Health Chatbot," *arXiv preprint arXiv:2411.14925*, 2024.

[15] C.-H. Tsai *et al.*, "Generating Personalized Pregnancy Nutrition Recommendations with GPT-Powered AI Chatbot. In: 20th International Conference on Information Systems for Crisis Response and Management," in *20th International Conference on Information Systems for Crisis Response and Management (ISCRAM)*, 2023, p. 263.

[16] K. M. Leung, Z. Liu, H. Y. Lam, Y. Zhang, W. T. Chan, and J. Lin, "Reinforcement Learning-Based Food Recommendation System for Dietary Optimization and Health Management," *Journal of Artificial Intelligence and Information*, vol. 2, pp. 108–112, 2025.

[17] M. M. Afsar, T. Crump, and B. Far, "Reinforcement learning based recommender systems: A survey," *ACM Comput Surv*, vol. 55, no. 7, pp. 1–38, 2022.

[18] X. Zhao, L. Xia, L. Zhang, Z. Ding, D. Yin, and J. Tang, "Deep reinforcement learning for page-wise recommendations," in *Proceedings of the 12th ACM conference on recommender systems*, 2018, pp. 95–103.

[19] Y. Li, M. Liu, J. Yin, C. Cui, X.-S. Xu, and L. Nie, "Routing micro-videos via a temporal graph-guided recommendation system," in *Proceedings of the 27th ACM international conference on multimedia*, 2019, pp. 1464–1472.

[20] V. Shetty *et al.*, "Role of ChatGPT in antenatal care," *World Journal of Advanced Research and Reviews*, vol. 23, no. 1, pp. 230–236, 2024.

[21] S. Balloccu, E. Reiter, V. Kumar, D. R. Recupero, and D. Riboni, "Ask the experts: sourcing high-quality datasets for nutritional counselling through Human-AI collaboration," *arXiv preprint arXiv:2401.08420*, 2024.

[22] D. Kurnianingtyas, N. Daud, I. Indriati, and L. Muflikhah, "Comparison Genetics Algorithm and Particle Swarm Optimization in Dietary Recommendations for Maternal Nutritional Fulfillment," *SITEKIN: Jurnal Sains, Teknologi dan Industri*, vol. 21, no. 2, pp. 216–227, 2024.

[23] A. Gharbi, "A dynamic reward-enhanced Q-learning approach for efficient path planning and obstacle avoidance in mobile robotics," *Applied Computing and Informatics*, 2024.

[24] M. Ganger, E. Duryea, and W. Hu, "Double Sarsa and double expected Sarsa with shallow and deep learning," *Journal of Data Analysis and Information Processing*, vol. 4, no. 04, p. 159, 2016.

[25] K. Godlewski and B. Sawicki, "Optimisation of MCTS Player for The Lord of the Rings: the card game," *arXiv preprint arXiv:2109.12001*, 2021.

[26] A. Khan *et al.*, "Adaptive filtering: issues, challenges, and best-fit solutions using particle swarm optimization variants," *Sensors*, vol. 23, no. 18, p. 7710, 2023.

[27] S. Moradi, "Comparative Study of Dynamic Optimization Techniques and Genetic Algorithms Approaches for Weight Reduction in Steel Truss Structures," 2025.

[28] L. Liu and J. T. Luo, "mctreesearch4j: A Monte Carlo Tree Search Implementation for the JVM," *J Open Source Softw*, vol. 7, no. 70, p. 3804, 2022.

[29] H. Zhang, H. Zhou, Y. Wei, and C. Huang, "Autonomous maneuver decision-making method based on reinforcement learning and Monte Carlo tree search," *Front Neurorobot*, vol. 16, p. 996412, 2022.