A Multi-Objective Multi-Agent Interactive Deep Reinforcement Learning Approach for Feature Selection

Rahma Hellali, Zaineb Chelly Dagdia, Karine Zeitouni

A Multi-Objective Multi-Agent Interactive Deep Reinforcement Learning Approach for Feature Selection

Rahma Hellali[1], Zaineb Chelly Dagdia[1], and Karine Zeitouni[1]

Université Paris-Saclay, UVSQ, DAVID, France

Abstract. Feature selection is paramount in data preprocessing for optimizing feature subsets in machine learning tasks. Intriguing recent developments in this area are the Interactive Reinforced Feature Selection methods, showing promise. However, they often prioritize singular metrics like accuracy, overlooking conflicting metrics or objectives. Moreover, their reliance on a single trainer can hamper efficiency by providing repetitive advice. To tackle these issues, this paper presents a Multi-objective Multi-agent Interactive Deep Reinforcement Learning method. We conceptualize feature selection as a multi-objective problem and introduce a novel reward assignment method that balances the Area Under the Curve (AUC) measure and the number of selected features. Furthermore, we propose a diverse trainer strategy that harnesses multiple trainers to prevent repetitive guidance. This strategy leverages diverse external trainers for accelerated feature exploration and fosters self-exploration by the agent. Our approach yields Pareto Front solutions, offering flexibility for decision-makers in selecting the final optimal feature set. Empirical results demonstrate superior performance compared to state-of-the-art feature selection methods. Additionally, through the Pareto Front solutions, our method effectively manages the trade-offs between AUC and feature count.

Keywords: Feature selection · deep reinforcement learning · multi-agent systems · multi-objective optimization.

1 Introduction

Feature Selection (FS) entails identifying a subset of the most pertinent features for inclusion in a machine-learning model. This can enhance the performance of a model thus mitigating overfitting and enhancing generalization. Recently, reinforced FS methods [8,2] have emerged, treating FS as a Multi-Agent Deep Reinforcement Learning (MADRL) task; where each feature is associated with an agent responsible for selecting or deselecting its corresponding feature. In general, MADRL methods focus on optimizing systems with a single goal and finding a single solution. However, FS problems often have multiple conflicting objectives, resulting in multiple feasible solutions known as the Pareto Front (PF) set. These solutions effectively satisfy all objectives but are incomparable

to each other. To the best of our knowledge, no existing work treats the reinforced FS problem as multi-objective. In our approach, we refine this problem by formulating it as a multi-objective deep reinforcement learning, tackling two conflicting objectives: minimizing the number of selected features while maximizing the classification Area Under the Curve (AUC) score. Also, in MADRL, depending solely on self-exploration is insufficient to achieve both effectiveness and efficiency in feature exploration. Recently, Interactive Reinforcement Learning (IRL) has emerged as a promising approach, drawing inspiration from real-life biological learning scenarios and leveraging external knowledge. These IRL methods [7] have shown effectiveness in accelerating the exploration of Reinforcement Learning (RL). This interactive learning paradigm offers an exciting chance to incorporate teaching/guiding agents for more efficient exploration and learning in FS tasks. Recently, [2] proposed an IRL method based on external knowledge named trainers. However, their teaching strategy is employed for a specific duration of the learning process, which can be defined by a given number of iterations. Moreover, this strategy, based on a single trainer, tends to yield similar advice when similar active features are used as input. Also, this process remains static, as the same trainer continues to provide advice for a fixed period. To overcome these limitations, our approach additionally introduces a novel teaching strategy that enhances diversity and adaptability in providing advice to the agents using different trainers. This is achieved by selecting the best trainer advice in each iteration, ensuring that the agents receive more varied and effective guidance throughout the learning process.

In this paper, we introduce an efficient approach called Multi-Objective Interactive Deep Reinforcement Learning with Multiple Agents (MOIDRL-MA), tailored for feature selection. Our contributions can be summarized as follows: (1) We frame the FS challenge as a multi-objective problem—a novel approach not previously explored in the reinforced FS literature. The transition to a multi-objective framework is pivotal for informed decision-making. Our methodology presents a diverse array of solutions, striking a delicate balance between FS and classification performance. Decision-makers can select from these solutions according to their preferences and constraints, thereby enhancing adaptability in tackling real-world challenges. (2) We propose a multi-trainer teaching strategy to diversify external guidance by using multiple trainers. This approach also incorporates varying advice to agents, ensuring that repetitive guidance is minimized throughout the learning process, thereby enhancing the efficiency of MOIDRL-MA. (3) We showcase the effectiveness of our newly proposed MOIDRL-MA compared to RL and traditional FS methods, as well as the effectiveness of our newly proposed multi-trainer teaching strategy.

2 Related Works

Various studies have been carried out utilizing RL for feature selection. Authors in [5] introduced an RL approach based on a transformation graph that explores the feature space and an RL-driven process focused on optimizing performance

by identifying valuable features. The RL approach in [3] introduced an Average of Rewards criterion to evaluate the effectiveness of features based on their impact on state transitions. Additionally, an iterative algorithm is utilized to select the best subset of features using both filte and wrapper FS methods. In [6], a method employing multi-agent RL with a guiding agent was introduced. Each feature corresponds to a main and guide agent, jointly deciding feature selection. Main agents optimize feature selection using guide agent criteria, updating Q-values based on behavior differences. Authors in [9] introduced an RL single agent approach based on a reward and training levels interactive strategy to improve training efficiency with external advice. On the other hand, works on deep RL have been conducted focusing on MADRL for FS. Authors in [8] redefined FS within a deep RL framework, treating each feature as an individual agent. They acquired environment states through three distinct methods: statistical description, autoencoder, and graph convolutional network (GCN), enhancing the algorithm's comprehension of the learning process. Additionally, they investigated enhancing coordination among features by employing diverse metrics for computing rewards. Building upon [8], authors in [2] expanded the MADRL approach by incorporating a novel concept inspired by IRL. They introduced a hybrid teaching strategy, where different trainers take on the teaching role iteratively at different stages. This strategy enables agents to acquire a diverse range of knowledge. However, despite its iterative nature, the strategy may lead to redundancy, as a single trainer might offer similar advice. Furthermore, the hybrid process remains static, with the same trainer providing advice for a fixed number of episodes. As previously mentioned, MADRL approaches typically aim to optimize systems for a single objective, seeking a unique solution. It is noteworthy that despite the existence of Multi-Objective Deep Reinforcement Learning (MODRL) works, none have addressed the reinforced FS problem from a multi-objective perspective. In MODRL, each objective is associated with its reward signal, resulting in a reward vector instead of a single numerical value. Actions are selected based on this reward vector to attain optimal solutions. Various algorithms have been proposed to find these optimal solutions, categorized into two groups based on the number of optimal policies they identify [4]: (1) single-policy algorithms, which aim to find the best solution closest to the optimal policy, determined by preferences among objectives defined according to the problem domain or user feedback, and (2) multiple-policy algorithms, which are geared toward discovering a collection of optimal solutions that approximate the Pareto Front set. To go beyond the state of the art and to mitigate the limitations of recent works, specifically [2], we introduce MOIDRL-MA that adopts multi-objective, interactive deep RL, multiple agents, and a multi-policy paradigm which enables agents with distinct policies to pursue unique strategies.

3 Proposed Method

The MOIDRL-MA pseudo-code is given in Algorithm 1, respectively. Initially, a feature agent is assigned to each feature slated for exploration (Algorithm 1, lines

10-16), tasked with determining its inclusion in the final selection. Before finalizing the selected feature subset that forms the environment, each agent receives guidance from trainers regarding whether to adjust its decisions based on past actions taken during previous episodes (Section 3.1). Specifically, for each trainer, a score is calculated based on the previously selected features (active agents). Subsequently, the adviser trainer will be identified as the one with the highest score. Advice will be provided solely to agents that opt to deselect their features (indecisive). The resultant feature subset becomes the environment in which feature agents engage in interactions (Algorithm 1, lines 17-30). In feature subspace exploration, the number of selected features varies resulting in a changing length of the state representation vector. Thus, we adopt the descriptive statistics state representation method described in [8] (Algorithm 1, line 9). Simultaneously, the actions carried out by feature agents contribute to a cumulative reward, which is then distributed among the participating agents. As MOIDRL-MA is built upon a multi-objective framework, we assess the overall reward by taking into account both AUC and the number of selected features in the downstream task (Section 3.2) (Algorithm 1, line 31). Our framework includes a control stage dedicated to training the agent, storing experiences, and accelerating the training process. To expedite training, each agent possesses a memory unit where tuples containing the state, action, reward, and next state are stored after each episode. Subsequently, each agent uses its corresponding Deep Q Network (DQN) to train its policies by randomly sampling mini-batches from this memory (Section 3.3). The agents iteratively update their policies based on the sampled experiences, gradually refining their decision-making capabilities and ultimately converging towards the selected final feature space (Algorithm 1, lines 34-37). The key components of our MOIDRL-MA are the following: **Multi-Agent:** we consider N features and create N agents each assigned to a specific feature and each of them trains a DQN. Each agent's role is to determine whether its corresponding feature should be selected or not. **Actions:** Agents have two actions: selecting or deselecting their features. **Environment:** It corresponds to the selected feature subset. When a feature agent takes action to either include or exclude a feature, the state of the feature subspace is accordingly modified. Additionally, the environment includes the actual AUC calculated based on the previously selected feature subset, along with the previously calculated AUC. These are crucial for reward assignment. **State:** The state s represents the selected feature subset. Our scenario presents a challenge as the state's length varies with each iteration, unlike in DQN scenarios where the input shape remains constant. To address this, we use meta-descriptive statistics outlined in [8] to derive the representation of s. **Reward:** The reward serves as an incentive for exploring the feature subspace. We evaluate the overall reward by considering the AUC score and the number of selected features subset in the downstream task and distribute it equally among agents that select features in the current step. **Trainer:** The trainer acts as the provider of guidance to the feature agents.

Algorithm 1 MOIDRL-MA Algorithm

Require: Number of Features: N, Epsilon value: ϵ, Learning Rate: γ, Number of Episodes: $Max\_episodes$, Number of trainer steps: $trainers\_steps$

for i **in** range(N) **do** #initialize the agents
  create $Agent^i$ models
end for
Vectors_list ← [0,0] * $Max\_episodes$
STATE = reset the environment by taking randomly a feature subset
previous_actions = [i for i in range(N)]
previous_AUCScore = calculate_AUC(previous_actions)
for $timestep$ **in** range($Max\_episodes$) **do**
  State_vectors = State_representation(State) #meta-descriptive
  for j **in** range(N) **do**
    if random value > ϵ **then**
      $Agent^j$ do Exploitation
    else
      $Agent^j$ do Exploration
    end if
  end for
  #Multi-Trainer Strategy
  active_agents = previous_actions
  initial_actions = selected_features(actions)
  if $timestep \geq$ Guides_steps **then**
    Confident_agents = Calculate_Confidents (active_agents, initial_actions)
    Indecisive_agents = Calculate_Indecisives (active_agents, initial_actions)
    for i **in** range($guides$) **do**
      $Score$, $advice$ = calculate_Guide(active_agents)
    end for
    Best_trainer_score = Max($Score$)
    selected_feature = $advice$
  else
    selected_feature = Intersection(active_agents, initial_actions)
  end if
  Reward, aucScore = calculate_reward(selected_feature, active_agents, previous_actions, previous_AUCScore) # Multi-Criteria Reward Allocation
  State_next, State_next_vector = Calculate_NextState(selected_feature)
  Vectors_list[$timestep$] = [aucScore, feature_number] #save all vectors
  for i **in** range(N) **do** # Experience replay and training process
    rewards_sample = Select_MiniBatch()
    Q_Value of $Agent^i$ = rewards_sample $Agent^i$ + $\gamma\times$ Future_Rewards
  end for
  STATE = STATE_next
  previous_actions = selected_feature
  previous_AUCScore = aucScore
end for
$Non\_dominated\_solutions$ = Pareto_Front(Vectors_list)
Return Non_dominated_solutions

3.1 Multi-Trainer Strategy

Addressing the concern of repetitive advice from a single trainer regarding similar input features, we draw inspiration from [2] and introduce a new teaching approach. This method utilizes FS algorithms to empower external trainers in effectively guiding agents. Central elements of our multi-trainer strategy involve:

Active/Inactive Features (Agents): To vary the feature inputs provided to trainers, we dynamically adjust inputs by selecting features known as "active features". These are features chosen by agents in the previous step. For example, if at step $t-1$, agents select f_2 and f_5, the active features at step t are f_2 and f_5. Agents associated with these active features are termed "active agents" F_a, while the remaining agents are considered as "inactive agents" who select or deselect their corresponding inactive features.

Confident/Indecisive Features (Agents): We dynamically categorize active features into: confident and indecisive features, linked with confident and indecisive agents, respectively. At each step t, features chosen by the agents are categorized as confident features, while those not chosen are termed indecisive features. For example, if at step t, confident features include f_2 and f_5, and agents agt_2 and agt_5 opt to select f_2 and deselect f_5, then f_2 becomes a confident feature with agt_2 as its corresponding agent, while f_5 is marked as an indecisive feature with agt_5 as its corresponding indecisive agent.

Advice and Initial Selections: Agents use their policy networks to make initial decisions at each step. They seek guidance from external trainers and adjust their actions accordingly, known as "advised actions." Consequently, only indecisive agents follow the advice and execute the advised actions, while confident agents retain their initial choices without modification based on trainer guidance.

Prior to calculating the indices of agents necessitating adjustments, we collect their inputs and evaluate a performance metric (e.g., F1) for each trainer based on the involved agents' performance (F_p). Our method allows for a customized set of trainers, such as K-best, LASSO, and Recursive Feature Elimination (RFE), defined by the user. At the current step t, we initialize the trainer set as $T = T_1, \ldots, T_n$. Each trainer T_i is trained on F_p to obtain its performance metric. Then, we determine the winning trainer, which advises the indecisive agents with the highest performance score. This ensures that in every iteration, the most effective guidance is selected from the trainer set T, thereby enhancing the teaching process and promoting better exploration. Specifically, during exploration, the presence of randomness may lead to the inclusion of similar features. Consequently, when similar features are present, a single trainer may offer comparable advice. To tackle this, our approach encourages diversified exploration by involving multiple trainers, each offering distinct advice. Furthermore, relying solely on guidance from a single trainer throughout the execution period may result in sub-optimal outcomes, potentially exposing agents to imprudent advice. By including guidance from multiple trainers with the highest scores, our method strikes a balance and mitigates the impact of poor advice. The self-learning ability of agents may diminish if they consistently rely on trainers' guidance. To

address this, agents gradually transition to independent exploration and learning after reaching half of the episode count, choosing to incorporate only the features deemed confident by the agents themselves (Algorithm 1, lines 17-30).

3.2 MODRL-based Strategy for Allocating Rewards

Our MOIDRL-MA builds upon the framework of MADRL. In our approach, each agent operates its own policy network to decide whether to include or exclude its corresponding features. Therefore, MOIDRL-MA adopts a multi-policy paradigm, where different agents are equipped with distinct policies $\pi_1, \ldots, \pi_N$, with N representing the number of agents, enabling them to pursue unique strategies or rules. To integrate the multi-objective paradigm into the MADRL FS problem and adapt our approach to MODRL, we incorporate preferences into the reward assignment function. Many commonly used criteria incorporate an AUC score measure adjusted by the penalty associated with the number of selected features [1]. MOIDRL-MA maintains control over both AUC and the number of features in each iteration, comparing them against those of the previous iteration. If the AUC score increases, a reward is equally assigned to all selected feature agents, regardless of the number of features. Conversely, if the AUC score decreases, a penalty is equally assigned to all feature agents, irrespective of the number of features. Consequently, when the AUC score remains the same as in the previous iteration, a penalty is assigned if the number of features increases; otherwise, all feature agents receive a reward. The penalty measure varies by method in multi-objective feature selection. While [10] uses a hyperparameter α multiplied by entropy, we chose to use the sum of information gain (IG) instead. This change was made because entropy overly diminishes the reward, potentially distorting reward allocation in RL. The sum of IG S_{IG}(input) is defined as:

$$S_{IG}(SF) = \sum_i IG(X_i; Y) \tag{1}$$

where SF is the set of selected features, $IG(X_i; Y)$ is the IG between feature X_i and the target variable Y. Thus, the penalty formula is expressed as:

$$\text{Penalty} = \text{AUC} - \alpha * S_{IG}(SF) \tag{2}$$

We record reward vectors for AUC and selected features during MOIDRL-MA execution. After the maximum episodes, we extract PF solutions from the accumulated training vectors (Algorithm 1, line 42).

3.3 Feature Subspace Exploration via Experience Replay

Experience Replay (ER) enhances RL efficiency. After each action, the most recent sample for agent i at episode t containing action (a_i^t), reward (r_i^t), current state (s_i^t), and next state (s_i^{t+1}) is stored in memory, replacing the oldest sample. Indecisive agents train their policies using ER by randomly selecting

mini-batches from this memory. Each agent trains its DQN with these mini-batches to maximize long-term rewards, as follows:

$$Q(s_i^t, a_i^t) = r_i^t + \gamma \; max \; Q(s_i^t, a_i^{t+1}) \tag{3}$$

where γ indicates the discount factor (Algorithm 1, lines 34-37).The algorithm persists until it reaches the maximum number of episodes.

4 Experimental Setup

We address the following Research Questions (RQ), utilizing the datasets outlined in Table 1: **RQ1:** How does our MOIDRL-MA, leveraging its Pareto Front solutions generated through our newly introduced multi-objective component, fare against existing reinforcement and traditional feature selection approaches? **RQ2:** How does our new teaching strategy enhance advice diversity for feature agents, and thereby improve the efficiency of MOIDRL-MA?

Table 1. Experimental datasets

Dataset	**# Features**	**# Instances**	**# Classes**
Wisconsin Breast Cancer (WBC)	32	569	2
Forest Cover Type (FCT)	54	15120	7
Spambase (Spam)	57	4601	2
Insurance Company Benchmark (ICB)	86	9822	2
Musk	166	6598	2
Toxicity	1203	171	2
Colon Cancer (Colon)	2000	62	2
Leukemia (LEU)	5148	72	2
ARCENE	10000	201	2
Mixed-lineage leukemias (MLL)	12534	72	3
LUNG	12601	203	5
Gene expression cancer RNA-Seq (RNA)	16383	801	5
TCGA Kidney Cancers (KIRC)	20532	1024	2

To address **RQ1**, we evaluated MOIDRL-MA against methods from the literature [6], [2], [8], [5], and [3] and four baseline algorithms: (i) mRMR, using the average number of features from MOIDRL-MA PF solutions; (ii) RFE, using the average number of features from MOIDRL-MA PF solutions; (iii) LASSO with a regularization weight $\lambda = 1.0$; and (iv) Genetic Feature Selection (GFS) with a crossover probability of 0.7, mutation probability of 0.1, and 50 generations. The comparison methodology consists of evaluating MOIDRL-MA against established RL feature selection approaches and widely used baseline algorithms to highlight its efficacy and contributions. For **RQ2**, we compared MOIDRL-MA to IRFS-HT [2], using meta-description statistics instead of GCN for state representation, with a mini-batch size of 64, AdamOptimizer with a learning rate of

0.01, a discount factor γ of 0.99, and a memory size of 2000 for experience replay. Our DQN consists of two ReLU layers with 64 and 8 nodes, respectively. We used 1000 episodes for the LEU, ARCENE, MLL, LUNG, RNA, and KIRC datasets, and 5000 for WBC, FCT, Spam, ICB, Musk, Toxicity, Colon, and ARCENE. Downstream tasks use logistic regression for WBC, random forest for FCT, ICB, Musk, and KIRC, XGBoost for Spam, Toxicity, MLL, and RNA, and SVM for Colon, LEU, ARCENE, and LUNG, with default parameters in scikit-learn. The data is randomly split into training data (70%) and test data (30%). All evaluations are conducted on a Python framework GPU with 128 GB RAM running a 64-bit Linux OS. All relevant implementation specifics and parameters can be accessible in our open-source code:https://github.com/HellaliRahma/MOIDRL-MA. In Tables 2 and 3, numbers in parentheses indicate selected features, except in the "Dataset" column where they represent initial features. For Table 2, only [6] and [3] provide the number of selected features. Table 2 compares datasets from literature solely based on Accuracy (%), with unreported results denoted by "-". Table 3 reports results based on the Area Under the Curve (AUC). In our approach, we use the k-best and decision tree trainers. Recalling that we can use T trainers, we use only two trainers to allow comparison with IRFS-HT. To calculate index adjustments for agents, we set: **K-best:** If the trainer finds an indecisive feature to be superior to at least half of the active features, then the corresponding agent should shift from deselecting to selecting that feature. We determine the number of confident features as $m = |F_c|$ and the number of indecisive features as $n = |F_i|$. Then, we set the integer $k = \lceil 2m + n \rceil$. Next, we employ K-best to select the top k features in F_a, denoted as F_{KBest}. The indices of agents requiring action adjustments encompass all indecisive agents $|F_i|$ belonging to F_{KBest}. **Decision tree:** We train it on F_a and obtain features' importance scores, denoted by $imp_{f_{a1}}, \ldots, imp_{f_{at}}$. For indecisive features F_i, their importance is defined as $IMP_i = imp_{f_{aj}} | f_{aj} \in F_i$; for confident features F_c, their importance is : $IMP_c = imp_{f_{aj}} | f_{aj} \in F_c$. We denote the median of IMP_c as g. The indices of agents requiring action adjustments encompass all indecisive agents $|F_i|$ that are superior to g.

5 Results and Discussions

Table 2 demonstrates that among the Pareto Front solutions, MOIDRL-MA consistently exhibits superior performance compared to other RL feature selection methods in most cases. In Colon, MOIDRL-MA achieves the highest accuracy. Among its 5 PF solutions, it achieves an accuracy of 95.8% with only 8 features. In contrast, [6] reported an accuracy of 94.7% with 38 features and 73% with 40 features for [3]. This significant contrast underscores the importance of MOIDRL-MA's components, rooted in deep learning, interactive RL, and multi-objective considerations. [6] relies on simple multi-agent RL, which may result in less accurate decision predictions compared to those achieved through deep learning techniques. Moreover, the approach proposed by [3] relies on a single agent, wherein each iteration requires the agent to iterate through 2^N possibil-

ities (actions), potentially leading to the attainment of only local optima. This contrasts with our MOIDRL-MA framework, where each agent makes simple decisions: selecting or deselecting features. In SPAM, among its 22 PF solutions, MOIDRL-MA outperforms [2] (93.3%). This superiority can be attributed to the strategy of diversifying the trainer in each episode, as opposed to maintaining the same trainer throughout all episodes. Such a diversified approach can significantly influence the training process, ultimately enhancing the algorithm's performance. Regarding the comparison with [3] (82.9%, 20 features), our approach also demonstrates superior performance due to its multi-agent component as explained above. [6] reported an accuracy of 96.3% with 20 features, and [5] reported an accuracy of 96.1%. In comparison to these, our approach still demonstrates comparable PF solutions, albeit with a slight decrease in accuracy to 93.2%, yet allowing for only 16 features, for instance. Regarding FCT, the PF solutions generated by our method do not reach the highest accuracy reported by [6] (88%); however, they perform better than [2] (80%) and are close to [8] (87.3%). We found solutions with slightly lower accuracy (86%) but a significant reduction in the number of features: 27 for MOIDRL-MA compared to 33 for [6]. This can be attributed to the importance of multi-objective flexibility in considering solutions that may still be relevant to decision-makers. Similarly, from Table 3, it can be observed that in almost all cases, among its PF solutions, MOIDRL-MA exhibited superior performance compared to the baseline methods; in terms of both objectives. For example, observing one PF solution, for Toxicity, our method achieved an AUC of 88.3% with 31 features, whereas mRMR registered an AUC of 61.5%, RFE reached 76.8%, GFS obtained 75.7% with 624 features, LASSO achieved 75.3% with 879 features, and the no-use scenario yielded to an AUC of 65.2%. Our MOIDRL-MA notably reduces the number of features from an initial 1203 to just 31 selected final features; with the highest AUC. Similarly for datasets with a larger number of features, MOIDRL-MA demonstrated superior performance. For instance, in KIRC, our method achieved an AUC of 91%, while mRMR 90.9%, RFE 81.8%, GFS 90.1% with 10185 features, LASSO 89.7% with 852 features, and the no use scenario registered an AUC of 81.8%. Moreover, our MOIDRL-MA remarkably reduces the number of features from an initial 20532 to just 17 final selected features. In conclusion, and to respond to **RQ1**, MOIDRL-MA, demonstrated promise in surpassing existing (deep) reinforcement and traditional FS approaches. By leveraging multi-objective optimization, we have successfully achieved PF solutions that provide decision-makers with a diverse range of options. This not only enhances the flexibility and adaptability of MOIDRL-MA but also underscores its practical utility in real-world scenarios. Additionally, while it is acknowledged that certain methods may occasionally outperform our approach in terms of Acc/AUC, a critical observation reveals the superiority of our method in substantially reducing the number of features in most cases. Table 3 illustrates the efficiency of our new teaching strategy compared to IRFS-HT. In the RNA dataset, our method achieves a comparable AUC of 99.8% among the PF solutions, while IRFS-HT achieves 99.7%. However, our approach notably reduces

the feature count to 63 features, while IRFS-HT employs a high number of 8207 features. In the KIRC dataset, our method shows strong AUC performance at 91%, compared to IRFS-HT's 89.7%. Furthermore, our approach utilizes only 17 features compared to IRFS-HT's notably high count of 10165 features. These results can be attributed to the integration of the concept of the winning trainer, which provides advice instead of relying solely on a single trainer. This approach significantly enhances the algorithm's performance by ensuring that the best advice, with the highest score, is used to adjust the agents' decisions. Moreover, by repeating this process in every episode instead of selecting specific intervals to change the trainer, the approach remains dynamic. This dynamism ensures that the provided advice evolves continuously throughout the execution process, contributing to the algorithm's adaptability and effectiveness. Moreover, IRFS-HT provides only one solution, identifying the solution with the highest AUC as a singular outcome. However, for the KIRC dataset, the obtained PF solutions reveal 17 non-dominated solutions with MOIDRL-MA. These solutions offer decision-makers various combination possibilities, underscoring the importance of carefully weighing priorities when selecting the most suitable solution for their needs. Figure 1 shows that MOIDRL-MA consistently outperforms IRFS-HT. MOIDRL-MA exhibits stable convergence at approximately 83% AUC from episode 750 onwards, while IRFS-HT remains unstable and fails to converge. Figure 2 highlights that MOIDRL-MA reduces the number of features more effectively than IRFS-HT. While IRFS-HT's feature count remains high and stable after episode 750, MOIDRL-MA reduces its feature count to under 1,000 by episode 750, demonstrating superior performance and decision-making through its teaching strategy. In conclusion, and to address **RQ2**, our MOIDRL-MA indicates promising potential in surpassing IRFS-HT, as evidenced by significant differences observed in feature reduction while maintaining high AUC performance. This endorses the effectiveness of our proposed new teaching strategy. A key challenge of the proposed method is its high memory and time consumption due to creating a DQN for each feature. This leads to significant computational costs as the number of features increases. To address this, we are currently developing a selective learning process that focuses on training only indecisive agents, aiming to reduce resource usage and execution time.

Table 2: MOIDRL-MA vs (deep) RL methods (Accuracy (%))

Dataset	# PF	MOIDRL-MA	[6]	[2]	[8]	[5]	[3]
WBC (32)	8	92.3 (1), 97.9 (4), 96.5 (2), 97.2 (3), 97.2 (3), 96.5 (2), 97.9 (5), 98.6 (9)	98.2 (17)	–	–	–	–
FCT (54)	17	85.4 (25), 85.2 (23), 85.4 (24), 85.7 (26), 54.7 (1), 85.1 (18), 84.0 (10), 75.1 (8), 87.1 (39), 69.8 (5), 68.9 (4), 59.3 (2), 71.1 (6), 77.7 (9), 86.2 (27), 86.7 (32), 72.7 (7)	88 (33)	80	87.3	–	–

Spam (57)	22	95.8 (33), 95.8 (37), 95.9 (41), 95.7 (29), 95.7 (32), 95.9 (39), 95.3 (25), 94.9 (24), 95.3 (26), 80.4 (1), 85.8 (3), 93.2 (16), 92.2 (10), 92.7 (15), 90.0 (9), 87.3 (4), 87.6 (6), 83.2 (2), 92.5 (13), 89.5 (8), 94.1 (20), 89.4 (7)	96.3 (20)	93	–	96.1	82.9 (20)
ICB (86)	20	90.8 (11), 95.4 (35), 92.0 (13), 93.5 (16), 95.6 (45), 90.8 (10), 95.9 (78), 95.9 (46), 94.4 (23), 64 (1), 93.3 (14), 91.8 (12), 94.9 (27), 95.6 (44), 89.1 (8), 94.2 (20), 94.1 (19), 95.3 (29), 89.1 (7), 83.8 (2)	94.3 (27)	91	–	–	–
Musk (166)	14	87 (5), 77.1 (2), 95.6 (32), 98.1 (62), 94.4 (17), 83.9 (4), 95.0 (21), 91.9 (12), 89.5 (6), 69.1 (1), 82.7 (3), 92.5 (13), 90.7 (8), 93.2 (14)	98.4 (30)	98	–	–	–
Colon (2000)	5	83.3 (2), 95.8 (8), 87.5 (7), 87.5 (4), 79.1 (1)	94.7 (38)	–	–	–	73 (40)
ARCENE (10000)	8	72 (3), 78 (7), 75 (5), 86 (129), 81 (20), 73 (1), 88 (867), 88 (406)	–	–	–	–	86 (70)

Table 3: MOIDRL-MA vs IRFS-HT vs classical FS algorithms (AUC (%))

Dataset	# PF	MOIDRL-MA	IRFS-HT	mRMR	RFE	GFS	LASSO	No Use
WBC (32)	8	92.3 (1), 97.9 (4), 96.7 (2), 97.3 (3), 97.3 (3), 96.7 (2), 98.1 (5), 98.7 (9)	98.6 (11)	97.6 (4)	93.9 (4)	97.8 (14)	97.2 (28)	94
FCT (54)	17	91.6 (25), 91.4 (23), 91.5 (24), 91.7 (26), 73.5 (1), 91.4 (18), 90.7 (10), 85.5 (8), 92.5 (39), 82.4 (5), 81.9 (4), 76.3 (2), 83.2 (6), 87 (9), 92 (27), 92.3 (32), 84.1 (7)	86.5 (34)	82.9 (16)	54.7 (16)	86.2 (32)	85.5 (52)	85.5
Spam (57)	22	95.9 (33), 95.9 (37), 96 (41), 95.8 (29), 95.8 (32), 95.9 (39), 95.3 (25), 95.0 (24), 95.4 (26), 80.5 (1), 86 (3), 93.2 (16), 92.3 (10), 92.7 (15), 90.1 (9), 87.4 (4), 87.7 (6), 83.3 (2), 92.5 (13), 89.6 (8), 94.1 (20), 89.5 (7)	96.2 (34)	93.1 (18)	93.9 (18)	95.2 (34)	96.5 (55)	95.4

ICB (86)	20	90.9 (11), 95.5 (35), 92 (13), 93.6 (16), 95.6 (45), 90.9 (10), 95.9 (78), 95.9 (46), 94.5 (23), 63.9 (1), 93.3 (14), 91.9 (12), 94.9 (27), 95.6 (44), 89.2 (8), 94.3 (20), 94.1 (19), 95.4 (29), 89.1 (7), 83.8 (2)	96.2 (68)	92.6 (23)	94.5 (23)	95.7 (55)	96.3 (83)	96.1
Musk (166)	14	87 (5), 77.3 (2), 95.7 (32), 98.2 (62), 94.5 (17), 83.9 (4), 95 (21), 92 (12), 89.4 (6), 69.7 (1), 82.7 (3), 92.6 (13), 90.8 (8), 93.2 (14)	95.6 (73)	82 (14)	80.8 (14)	92.8 (95)	87.6 (164)	88.2
Toxicity (1203)	9	91.4 (588), 84.2 (10), 90.1 (556), 88.5 (109), 75 (1), 81.1 (2), 83 (8), 88.3 (31), 84.3 (21)	86.9 (957)	61.5 (147)	76.8 (147)	75.7 (624)	75.3 (879)	65.2
Colon (2000)	5	85.7 (2), 96.4 (8), 89.3 (7), 87.9 (4), 77.9 (1)	95.8 (957)	70.8 (4)	66.6 (4)	91 (990)	95.8 (541)	83.3
ARCENE (10000)	8	73.5 (3), 78.2 (7), 75 (5), 86 (129), 81.3 (20), 72 (1), 87.8 (867), 87.8 (406)	87 (4968)	69 (178)	79 (178)	87 (5084)	70 (71)	83
LEU (5148)	4	80.4 (1), 97.1 (4), 87.0 (3), 84.1 (2)	96.5 (2489)	89.6 (4)	89.7 (4)	98.5 (2520)	93.1 (338)	96.5
MLL (12534)	9	92.8 (19), 97.7 (113), 95.4 (57), 95.2 (47), 95.1 (36), 90.8 (7), 85.9 (3), 76.1 (1), 76.1 (1)	96.1 (6144)	88.4 (32)	92.3 (32)	98.1 (6205)	84.6 (500)	92.3
LUNG (12600)	13	98.6 (23), 97.4 (12), 91.3 (3), 85.3 (2), 92.3 (5), 96.2 (8), 98.0 (14), 71.3 (1), 71.3 (1), 99.7 (37), 98.3 (18), 95.1 (7), 99.1 (24)	99 (6142)	97.6 (17)	98 (17)	99.1 (6263)	99.5 (1365)	95.1
RNA (16383)	19	99.7 (55), 96.1 (10), 99.1 (31), 98.1 (21), 96.9 (12), 95.6 (8), 94 (6), 74.1 (2), 67.3 (1), 99.2 (32), 91.7 (4), 88.5 (3), 99.8 (63), 97.9 (15), 99.8 (102), 99.2 (35), 99.6 (47), 98.9 (23)	99.7 (8207)	99.7 (30)	99.3 (30)	99.7 (8253)	99.7 (2910)	99.5

KIRC (20532)	17	94.7 (493), 93.5 (210), 97.9 (4525), 85.7 (13), 81.1 (9), 85.4 (10), 70 (2), 95.7 (3952), 78.5 (7), 86.8 (16), 91.3 (70), 66.1 (1), 75.5 (3), 81.1 (9), 92.6 (92), 91 (17), 77.6 (4), 98.9 (23)	89.7 (10165)	90.9 (555)	81.8 (555)	90.1 (10185)	89.7 (852)	81.8

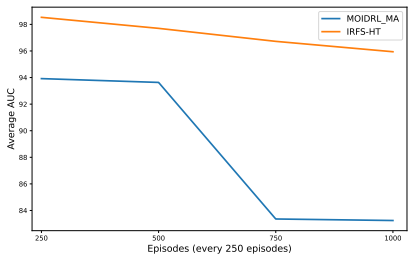

Fig. 1. The AUC performance per episode on the MLL dataset

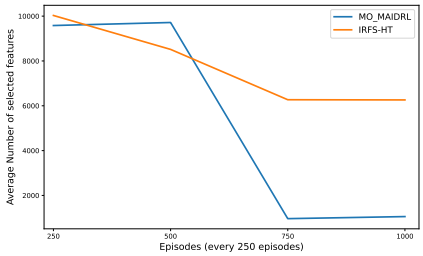

Fig. 2. The count of selected features per episode on the MLL dataset

6 Conclusion

MOIDRL-MA integrates interactive deep reinforcement learning with multi-objective optimization, excelling in feature selection by leveraging Pareto Front solutions to balance AUC and feature count, offering experts tailored options. Its key strength lies in generating customized solutions, but it incurs high memory

and time costs due to numerous DQN-based agents. Future work includes implementing selective learning for indecisive agents to mitigate resource constraints, applying the method to multiomics data in the ANR project RECORDS, encompassing metabolomic and transcriptomic datasets with up to 69,000 features, and developing Pareto Front solutions for sepsis diagnosis, optimizing both accuracy and biomarker count.

Acknowledgments

This study received funding from the "Programme d'Investissements d'Avenir (PIA)" under the France 2030 program (ANR-18-RHUS-0004) and is part of the Federation Hospitalo-Universitaire (FHU) Saclay and Paris Seine Nord SEPSIS initiative. It was also supported by ANR PIA funding (ANR-20-IDEES-0002). This work was performed using HPC resources from GENCI-IDRIS (Grant 2023-AD011014619); and also HPC Grid'5000.

References

1. Cavanaugh, J.E., Neath, A.A.: The akaike information criterion: Background, derivation, properties, application, interpretation, and refinements. Wiley Interdisciplinary Reviews: Computational Statistics **11**(3), e1460 (2019)
2. Fan, W., Liu, K., Liu, H., Wang, P., Ge, Y., Fu, Y.: Autofs: Automated feature selection via diversity-aware interactive reinforcement learning. In: 2020 IEEE International Conference on Data Mining (ICDM). pp. 1008–1013. IEEE (2020)
3. Fard, S.M.H., Hamzeh, A., Hashemi, S.: Using reinforcement learning to find an optimal set of features. Computers & Mathematics with Applications **66**(10), 1892–1904 (2013)
4. Hayes, C.F., Rădulescu, R., Bargiacchi, E., Källström, J., Macfarlane, M., Reymond, M., Verstraeten, T., Zintgraf, L.M., Dazeley, R., Heintz, F., et al.: A practical guide to multi-objective reinforcement learning and planning. Autonomous Agents and Multi-Agent Systems **36**(1), 26 (2022)
5. Khurana, U., Samulowitz, H., Turaga, D.: Feature engineering for predictive modeling using reinforcement learning. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 32 (2018)
6. Kim, M., Bae, J., Wang, B., Ko, H., Lim, J.S.: Feature selection method using multi-agent reinforcement learning based on guide agents. Sensors **23**(1), 98 (2022)
7. Lin, J., Ma, Z., Gomez, R., Nakamura, K., He, B., Li, G.: A review on interactive reinforcement learning from human social feedback. IEEE Access **8**, 120757–120765 (2020)
8. Liu, K., Fu, Y., Wang, P., Wu, L., Bo, R., Li, X.: Automating feature subspace exploration via multi-agent reinforcement learning. In: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. pp. 207–215 (2019)
9. Liu, K., Wang, D., Du, W., Wu, D.O., Fu, Y.: Interactive reinforced feature selection with traverse strategy. Knowledge and Information Systems **65**(5), 1935–1962 (2023)
10. Zheng, K., Wang, X.: Feature selection method with joint maximal information entropy between features and class. Pattern Recognition **77**, 20–29 (2018)