Feature and Instance Joint Selection: A Reinforcement Learning Perspective

Wei Fan[1] , **Kunpeng Liu**[2] , **Hao Liu**[3] , **Hengshu Zhu**[4] , **Hui Xiong**[5]* and **Yanjie Fu**[1]*

[1]University of Central Florida
[2]Portland State University
[3]Hong Kong University of Science and Technology
[4]Baidu Talent Intelligence Center
[5]Rutgers University

weifan@knights.ucf.edu, kunpengliu0827@gmail.com, liuh@ust.hk, zhuhengshu@gmail.com,
hxiong@rutgers.edu, yanjie.fu@ucf.edu

Abstract

Feature selection and instance selection are two important techniques of data processing. However, such selections have mostly been studied separately, while existing work towards the joint selection conducts feature/instance selection coarsely; thus neglecting the latent fine-grained interaction between feature space and instance space. To address this challenge, we propose a reinforcement learning solution to accomplish the joint selection task and simultaneously capture the interaction between the selection of each feature and each instance. In particular, a sequential-scanning mechanism is designed as action strategy of agents and a collaborative-changing environment is used to enhance agent collaboration. In addition, an interactive paradigm introduces prior selection knowledge to help agents for more efficient exploration. Finally, extensive experiments on real-world datasets have demonstrated the improved performances.

1 Introduction

Data preprocessing is to make input data most appropriate for model training. Generally, two well-known preprocessing techniques are feature selection [Liu and Motoda, 2012] and instance selection [Brighton and Mellish, 2002]. Feature selection is to select most important features to increase predictive performance (e.g., accuracy) and reduce feature size. Instance selection shares the similar objective to simultaneously improve modeling accuracy and decrease instance size.

In prior literature, feature selection and instance selection are usually regarded as two separate problems. Limited algorithms have been proposed for their joint selection. The joint feature and instance selection was initially addressed using genetic algorithms [Kuncheva and Jain, 1999]. Then some studies solve this problem with principal component analysis [Suganthi and Karunakaran, 2019] or application of pairwise similarity [Benabdeslem *et al.*, 2020]. In fact, the selection of each feature and each instance are mutually influenced

and jointly decide the final selection results. However, most previous work of joint selection conducts feature/instance selection coarsely and neglects the fine-grained interaction of feature space and instance space, which largely hinders the predictive performances on the selected data.

Reinforcement Learning (RL), as an effective tool, has great potential to learn the optimal results for search problems like such selections [Liu *et al.*, 2019]. In particular, different agents (i) individually take actions step by step that is appropriate to model each fine-grained selection choice; (ii) mutually interact with each other that can capture the interaction between feature space and instance space; (iii) jointly targets for optimal decisions that can be regarded as the selected data of joint selection task. To this end, we make a first attempt to leverage reinforcement learning for the joint feature instance selection problem. For this goal, several challenges arise.

First, how to formulate the joint feature instance selection task with reinforcement learning? Feature/Instance selection usually repeats two steps: select a subset and test the performance. If viewed from the RL perspective, this exploratory process is an agent first selects features/instances and then observe the selected data to get reward. Considering the two selections, we naturally reformulate the joint selection with a dual-agent reinforcement learning paradigm. Specifically, we create two RL agents: 1) a feature agent aims to select the optimal feature subset; 2) an instance agent aims to select the optimal instance subset. The two agents perceive selected features and instances as the state of environment, collect data characteristics as reward, and interact with each other to search for optimum selection results.

Second, how can we enable the two agents to simultaneously and collaboratively conduct joint selection? On one hand, if the feature or instance agent selects a subset each time, the agent needs to make n binary selections on n features/instances. This results into action space is exponentially-increasing (2^n) with the number of features/instances, leading to practical implementation difficulties. Thus, we propose a *sequential-scanning* mechanism for action design of agents. Specifically, we organize the selection decisions of features as a sequence and let the feature agent iteratively scan over this sequence to (de)select one feature each time. The instance agent adopts the same scan-

---
*Corresponding Author

ning strategy. This mechanism significantly reduces action space from exponential (2^n) to binary choice and transforms selection as a sequential decision-making process, which is appropriate for RL. On the other hand, the performance of a downstream model highly depends on the joint quality of both features and instances. To achieve the global optimal of joint selection, the two agents need to collaborate to learn the mutual influence of features and instances. We thus develop a *collaborative-changing* environment for agents. We regard the environment as the selected data sub-matrix, where the columns (features) and rows (instances) are simultaneously changed by the actions of dual agents. This shared environment captures actions of the two agents and jointly sense the data quality in two dimensions.

Third, how can the two agents learn prior knowledge to improve learning efficiency? Interactive RL [Amir *et al.*, 2016] has shown its superiority on speeding up agent exploration by learning from human experts or prior knowledge. In this regard, we utilize two external trainers to teach the two agents respectively via interactive RL: we introduce 1) a random forest based trainer with knowledge of feature importance to teach feature agent to pick features. 2) an isolation forest based trainer to recognize instance anomaly to teach instance agent how to filter out instances. With the advice of the two trainers, the two agents can learn the patterns of spotting and selecting quality features and instances more efficiently.

In summary, we propose a dual-agent interactive reinforcement learning framework to model the interaction of the joint feature and instance selection. Our contributions are: (i) we formulate the joint selection task with dual-agent reinforcement learning; (ii) we propose a sequential-scanning mechanism and a collaborative-changing environment to achieve the simultaneous and interactive selection; (iii) we leverage interactive reinforcement learning to improve the learning efficiency; (iv) we conduct extensive experiments to demonstrate our improved performances.

2 Backgrounds

Concept 2.1 Dual-Agent Reinforcement Learning. As a variant of multi-agent RL, dual-agent RL has two agents collaboratively accomplish two different tasks. For example, it has been applied to interactively generate bounding boxes and detect facial landmarks in computer vision [Guo *et al.*, 2018].

Concept 2.2 Interactive Reinforcement Learning. It is to provide agents with action advice from teacher-like trainers, so that agents learn learn the optimal decisions more efficiently in early exploration [Amir *et al.*, 2016].

Concept 2.3 Feature Selection. Given an input data matrix $X \in \mathbb{R}^{n*m}$, n and m denote the number of instances and features respectively; x_{ij} is the element in the i-th row and j-th column. We denote input features $F = \{\mathbf{f}_j\}_{j=1}^m$, where the j-th feature is denoted by $\mathbf{f}_j = \{x_{1j}, x_{2j}, ..., x_{nj}\}$. Feature selection aims to select an optimal subset $F^* \subseteq F$ making downstream predictive model perform well.

Concept 2.4 Instance Selection. As we mentioned above, for the data matrix X, all the instances are denoted as $C =$ $\{\mathbf{c}_i\}_{i=1}^n$ where the i-th instance $\mathbf{c}_i = \{x_{i1}, x_{i2}, ..., x_{im}\}$. Traditionally, instance selection studies aim to remove data noise or outliers, and find an instance subset $C^* \subseteq C$ that has the same performances as C for the downstream predictive task [Wilson and Martinez, 2000].

Definition 2.5 Feature and Instance Joint Selection. This task is to simultaneously find the optimal feature subset F^* and the optimal instance subset C^*, in order to achieve the best performances in a downstream predictive task.

3 Method

We address aforementioned challenges and propose a framework which called Dual-Agent Interactive Reinforced Selection (DAIRS), to model joint feature and instance selection task and introduce prior selection knowledge to agents.

3.1 DAIRS

As a reinforcement learning based framework, the DAIRS framework consists of dual agents, actions, states, reward and trainers. Specifically,

Dual Agents
The two agents are: the *Feature Agent* which models feature-feature correlation to select an optimized feature subset; the *Instance Agent* which models instance-instance correlation to select an optimized instance subset. However, the local optimal in feature or instance selection can't guarantee the global optimal of joint feature-instance selection. As a result, the two agents need to strategically collaborate with each other.

Actions
The dual agent action design is critical, because we need to consider: (i) the action space that determines the computational complexity and the learning efficiency of agents; (ii) the fine-grained interaction between feature space and instance space. The two considerations make it impossible to directly apply classic multi-agent reinforced selection [Liu *et al.*, 2019] to joint features and instance selection. To tackle this challenge, we develop a *sequential-scanning* with restart mechanism (Figure 1). Specifically, *Actions of the feature agent* are to sequentially scan all the m features and then restart scanning of the features over steps, where each step will select or deselect one feature. Let us denote a feature action record by $\mathbf{a}_F = \{a_i\}_{i=1}^m$, where the i-th action $a_i = 1$ or $a_i = 0$ means to select or deselect the i-th feature. Since the feature agent will restart the sequential scanning of m features, at Step t, the feature agent decides to select or deselect $\mathbf{f}_{t\,(mod\,m)}$. *Actions of the instance agent* are to sequentially scan all the n instances, and then restart scanning of the instances over steps, where each step will select or deselect one instance. Let us denote an instance action record as $\mathbf{a}_I = \{a_i\}_{i=1}^n$, where the i-th action $a_i = 1$ or $a_i = 0$ means to select or deselect the instance $\mathbf{c}_i$. At Step t, the instance agent decides to select or deselect instance $\mathbf{c}_{t\,(mod\,n)}$.

The sequential scanning strategy allows features and instances to be simultaneously selected or deselected. The dual interaction will coordinate the scanning actions of the two agents to generate a globally optimized data subspace of instances and features.
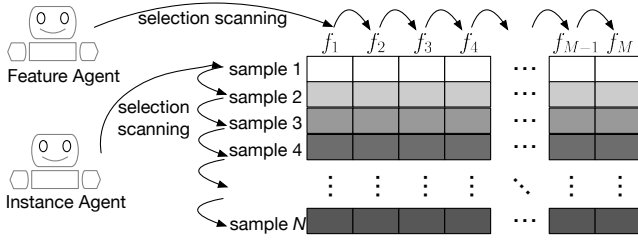
Figure 1: Actions of dual agents with sequential scanning. Agents iteratively scan over each feature or each instance to make the select/deselect decisions.

Figure 2: State of collaborative-changing environment, coordinating with actions of the feature agent and the instance agent.

State of the Environment

Instead of separately creating two environments for feature agent and instance agent, we develop a sharing environment to support simultaneous interaction between agents. The state is to quantitatively represent the situation of the *collaboratively-changing* environment. However, this is a non-trivial task: at the t-th step, the action records of the feature and instance agents, denoted by $\mathbf{a_F}^t$ and $\mathbf{a_I}^t$, can respectively derive a feature subset F' and an instance subset C'. The two subsets jointly form a sub-matrix of the input data X. The challenge is that the selected sub-matrix X' cannot directly be regarded as the state, because its dimensions change dynamically over time, while learning the policy networks of the dual agents require a fixed-length state representation.

To address this challenge, we develop a dynamic state representation method inspired by the image processing technique [Lu and Weng, 2007]. Specifically, by regarding the selected data sub-matrix as a 2-dimensional image, we first fix the state dimensions by padding the deselected positions with the padding token (zero). Formally, for the input data $X \in \mathbb{R}^{n*m}$, at the t-th step, $\mathbf{h}^t$ is measured by:

$$\mathbf{h}^t = \underbrace{\left[\mathbf{a_I}^{tT}||\cdots||\mathbf{a_I}^{tT}\right]}_{m} \otimes X \otimes \underbrace{\left[\mathbf{a_F}^{tT}||\cdots||\mathbf{a_F}^{tT}\right]}_{n}^{T} \quad (1)$$

where $\otimes$ is element-wise product, ^{T} is the transpose of a given matrix, $\mathbf{a_I}^t$ and $\mathbf{a_F}^t$ are the action records of feature agent and instance agent at the step t; m is the number of features; n is the number of instances. We then utilize a single convolutional layer to output the final representation. Formally, the state representation at Step t is computed by:

$$\mathbf{s}^t = Conv(\mathbf{w_s}\mathbf{h}^t + \mathbf{b_s}) \quad (2)$$

where $\mathbf{w_s}$, $\mathbf{b_s}$ is tuning weight parameters and bias, and $Conv$ is the convolution operation. Figure 2 shows the state representation process of our proposed collaborative-changing environment for dual agents.

Reward

The reward r is to inspire the exploration of the feature agent and instance agent. Since the actions of dual agents are sequential scanning, we measure the reward based on the characteristic difference between last state and current state, in order to temporally train reinforcement learning decision-making process [Sutton and Barto, 2018]. Specifically, sup-
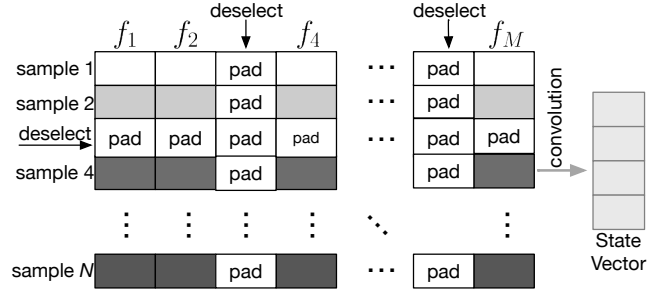
posing there are metrics set $\mathcal{K}$ we are caring about, we measure the performance difference Δ_i at Step t:

$$\Delta_i^t = k_i^t - k_i^{t-1} \quad (3)$$

where k_i^t means the i-th metric of the selected data subset at Step t. Then, the overall reward r at Step t is measured by:

$$r^t = \frac{1}{|\mathcal{K}|} \sum_{k_i \in \mathcal{K}} \Delta_i^t. \quad (4)$$

The actions of feature agent and instance agent collaboratively change the state, which directly determine the reward measurement. Then the reward is shared between the two agents to inspire exploration.

Trainers

We introduce the concept of teacher-like trainers from interactive reinforcement learning into our framework. We develop a random forest trainer for the feature agent and an isolation forest trainer for the instance agent. The two trainers can guide agents to explore better selection policies.

3.2 Model Training

Figure 3 shows an overview of our framework. Dual agents have their own Deep Q-Networks (DQNs) [Mnih *et al.*, 2013] as action policies. Two external trainers respectively guide the feature agent and the instance agent for more efficient exploration. The actions of the dual agents are to sequentially scan features and instances, which collaboratively decide the selected data sub-matrix X'. Then the state of the environment is derived from the sub-matrix and the reward is collected to inspire both feature agent and instance agent to select data.

To evaluate the policy, we use feed-forward neural network to approximate the Q value. The deep Q-learning iteratively optimizes the following loss function:

$$\mathbb{E}(r + \gamma \max_{a'} Q_{\theta'}(s', a') - Q_\theta(s, a))^2 \quad (5)$$

where r is the reward of action a in state s; a' is the next action in next state s'; γ is a discount factor; $Q_{\theta'}$ is the target network. For the function approximation, the Q function is with parameter θ. The gradient descent is:

$$\theta_{t+1} \leftarrow \theta_t + \alpha(r + \gamma \max_{a'} Q_{\theta'}(s', a') - Q_{\theta^t}(s, a)) \quad (6)$$

where α is the learning rate and t is the training step. By optimizing θ, the action policies of agents are constructed in order to maximize the long-term reward.
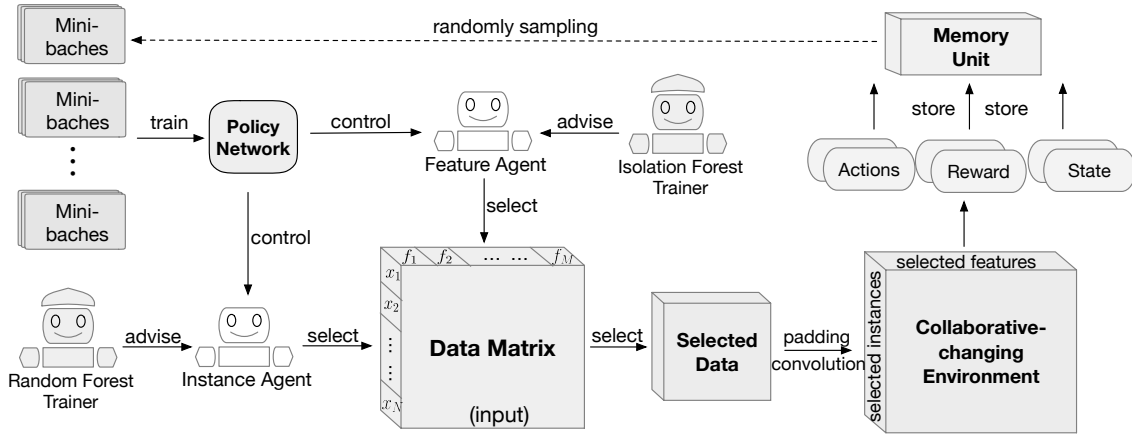
Figure 3: Framework Overview. Two agents collaboratively act and learn. Two trainers come to advise to help training.

3.3 Guiding Dual Agents via Interactive Reinforcement with External Trainers

Figure 3 shows how we leverage the prior knowledge of external trainers (i.e., classical feature selection and instance filtering methods) to guide feature agent and instance agent to improve their learning efficiency.

Guiding Feature Agent with Random Forest Trainer

Random forest classifier [Pal, 2005] can learn a set of decision trees to measure feature importance. We propose a trainer, namely random forest trainer, for the feature agent. The intuition is that feature importance can provide decision making support to the feature agent: if the trainer found a feature important, it would be suggested to select the feature; if not important, it would be suggested to deselect the feature. These advice can make the feature agent more aware of characteristics of feature space. We develop a three-step algorithm as follows:

First, we train a random forest classifier on the given m features $\{\mathbf{f}_1, \mathbf{f}_2, ..., \mathbf{f}_m\}$, to obtain the importance of each feature, denoted by $\{imp_1, imp_2, ..., imp_m\}$.

Second, based on the feature importance, we design a selection probability distribution $\mathbf{p_{RF}}\{p_1, p_2, ..., p_m\}$ for the m features, where the probability for i-th feature is given by:

$$p_i = \begin{cases} 1 & imp_i > \frac{\beta}{m} \\ m * imp_i & imp_i \le \frac{\beta}{m} \end{cases} \quad (7)$$

where β is a parameter to control suggested features.

Third, based on the probability, we sample an advised action list each step, denoted by $\mathbf{a_{RF}}$. The feature agent follows $\mathbf{a_{RF}}$ to take actions at the beginning steps of exploration.

Guiding Instance Agent with Isolation Forest Trainer

We aim to leverage the ability of external trainers for recognizing noisy or perturbing data samples to provide advice to instance agent for instance selection. Our underlying intuition is that bad instances differ from normal instances in the feature space [Aggarwal and Yu, 2001], and an agent can follow how classic instance filtering methods identify them.

We propose to identify bad-performed instances via an outlier detection algorithm and let instance agent try to rule out these points. Based on isolation forest [Liu *et al.*, 2008], an outlier detector, we propose another trainer called isolation forest trainer. We show how this trainer gives advice to the instance agent step by step:

First, we first group instances $\{\mathbf{c}_1, \mathbf{c}_2, ..., \mathbf{c}_n\}$ based on their labels $\mathbf{Y}\{y_1, y_2, ..., y_n\}$ in classification tasks. We denote these instance groups by $G = \{\mathcal{G}_{l_1}, \mathcal{G}_{l_2}, ..., \mathcal{G}_{l_p}\}$, where distinct labels $\{l_1, l_2, ..., l_p\} = set(\mathbf{Y})$ and the group towards k-th distinct label l_k is by $\mathcal{G}_{l_k} = \{\mathbf{c}_i \mid y_i = l_k \ \& \ y_i \in \mathbf{Y}\}$.

Second, for each group in G, we utilize the isolation forest algorithm to detect and filter outlier points. The filtered results are by $G_{IF} = \{IF(\mathcal{G}_{l_1}), IF(\mathcal{G}_{l_2}), ..., IF(\mathcal{G}_{l_p})\}$, where IF is the filtering operation with isolation forest.

Third, we derive the advised action list $\mathbf{A_{IF}}$ from filtered results G_{IF}. Specifically, for $\mathbf{a_{IF}}$ $\{a_1, a_2, ..., a_n\}$, the i-th advised action $a_i = 1$ if $\mathbf{c}_i \in G_{IF}$ and $a_i = 0$ if $\mathbf{c}_i \notin G_{IF}$. At several beginning steps of the exploration in reinforcement learning, the instance agent follows the advice and take the advised actions for better training.

4 Experiment

4.1 Experimental Setup

Datasets and Metrics. We use four public datasets of different domains on classification task to validate our methods: *ForestCover (FC)* dataset is a publicly available dataset from Kaggle[1] including characteristics of wilderness areas. *Madelon* dataset is a Nips 2003 workshop dataset containing data points grouped in 32 clusters and labeled by 1 and -1 [Dua and Graff, 2017]. *Spam* dataset is a collection of spam emails [Dua and Graff, 2017]. *USPS* dataset is a handwritten digit database including handwritten digit images [Cai *et al.*, 2010]. We report *Accuracy* and *F1-score* of certain downstream models to show the quality of selected data.

Baseline Algorithms. We compare predictive performance of our proposed model with different baselines, including instance selection methods, feature selection methods and feature instance joint selection methods: **DROPs** includes different instance reduction algorithms where we take DROP1 and

---

[1]https://www.kaggle.com/c/forest-cover-type-prediction/data

Dataset	FC		Madelon		Spam		USPS	
Model	Accuracy	F1-score	Accuracy	F1-score	Accuracy	F1-score	Accuracy	F1-score
DROP1	58.928	59.477	51.923	52.048	89.572	89.531	86.559	86.464
DROP5	64.682	64.975	52.435	52.431	89.717	89.718	91.612	91.607
GCNN	61.419	61.460	53.717	53.732	90.803	90.846	93.570	93.585
LASSO	61.684	62.705	54.230	54.223	90.658	90.699	91.899	91.937
RFE	65.828	66.430	55.256	55.261	88.776	88.862	93.763	93.786
LS2AOD	65.057	65.665	53.974	53.983	90.948	90.978	93.293	93.343
GeneticFSIS	65.834	66.033	55.938	55.837	90.742	90.675	93.242	93.321
IFS-CoCo	66.102	65.723	58.384	58.323	91.042	91.021	93.417	93.519
sCOs	66.213	66.104	57.992	57.976	90.810	90.632	93.792	93.738
DAIRS (Ours)	**67.328**	**66.908**	**61.282**	**61.281**	**91.745**	**91.389**	**94.122**	**94.071**

Table 1: Predictive performance of different selection methods with logistic regression as the downstream model.

Dataset	FC	Madelon	Spam	USPS
Feature	0.8703	0.5980	0.8771	0.7187
Instance	0.7483	0.8829	0.6770	0.6266

Table 2: Selection ratio on different datasets.

Figure 4: Performance of DAIRS variants on Madelon dataset.

DROP5 as baselines [Wilson and Martinez, 2000]. **GCNN** [Chou *et al.*, 2006] proposes the weak criterion employed by Condensed Nearest Neighbor. **LASSO** [Tibshirani, 1996] conducts feature selection and shrinkage via $l1$ penalty. **RFE** (Recursive Feature Elimination) recursively deselects the least important features. **LS2AOD** selects features via Laplacian Score and then samples via AOD. **GeneticFSIS** [Tsai *et al.*, 2013] applies genetic algorithms to alternatively select features and instances. **IFS-CoCo** [Derrac *et al.*, 2010] applies the cooperative co-evolutionary algorithm to select features and instances. **sCOs** [Benabdeslem *et al.*, 2020] uses a similarity preserving for co-selection of features and instances. For evaluation, we use these algorithms to select features/instances to get the data subset, and evaluate the quality of selected data towards prediction.

Implications. To compare fairly, we set the downstream task as a simple classifier logistic regression. The downstream task takes selected data as input and output the classification results. We randomly split the data into train data (70%) and test data (30%) where categorical features are encoded in one-hot. Baseline models are set the same selection ratio as our model. The size of memory unit is set to 300 in experience replay. For the reward measurement, we consider accuracy, relevance score and redundancy score following [Liu *et al.*, 2019]. The policy networks are set as two fully-connected layers of 512 middle states with ReLU as activation function. In RL exploration, the discount factor γ is set to 0.9, and we use ϵ-greedy exploration with ϵ as 0.8.

4.2 Overall Performances

Table 1 shows the overall predictive performance of logistic regression on the data selected by our proposed DAIRS and compared baselines. We report on four different datasets with respect to accuracy and f1-score and can observe our proposed model outperforms other selection methods, which signifies the best quality of our selected data for downstream predictions. Apart from our model, we notice other joint
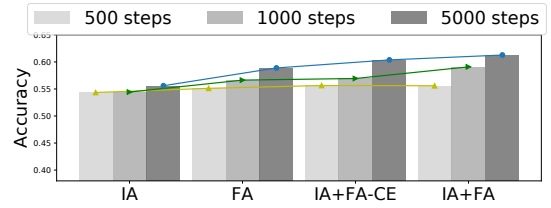
selection methods (e.g., IFS-CoCo) can have better performance than other baselines, for these methods consider both two dimensions (feature selection and instance selection).

Accordingly, as it shows in Table 2, our model can also acquire the best selection ratio to achieve higher predictive performance in exploration, while many other methods (e.g., DROP, RFE) cannot automatically learn the ratio that needs to be pre-defined. This demonstrates the superiority of our methods for automating to select the best data without handcrafted tunning. Also, it is easy to find out in most cases, most features and instances are useful for accurate predictions, while a few ones disturb predictions.

4.3 Study of Dual-agent Reinforced Selection

We aim to study the impacts of different components in our dual-agent reinforced selection framework. We consider four variant methods: (1) *FA* removes the instance agent, only considers the Feature Agent and creates environment for selected features as [Liu *et al.*, 2019]. (2) *IA* removes the feature agent and only considers Instance Agent. (3) *IA+FA-CE* removes Collaborative-changing Environment and creates an independent environment for each agent. (4) *IA+FA* is our proposed DAIRS model with two agents. Figure 4 shows performance comparison of these variants on Madelon dataset. Compared to single agent's exploration, we can easily observe a performance gain by dual agents.

Moreover, while both features and instances decide quality of the selected data, feature agent is more influential on predictions than instance agent when single agent explores. This suggests feature's quality is more important in data selection. We also find out the sharing environment is actually important for dual-agent coordination, which can make for better performance. From both figures, we observe the trend
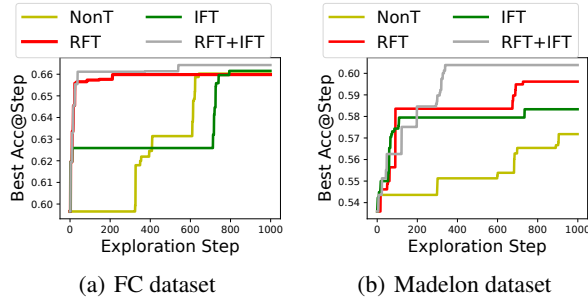
(a) FC dataset   (b) Madelon dataset

Figure 5: Exploration efficiency of variants with different interactive reinforced trainers.

(a) FC dataset   (b) Spam dataset

Figure 6: Visualizations of agent exploration process on FC and Spam dataset. Points in red are top 10 best performed steps.

of blue line is more significant than yellow line; this shows our method achieves better results in long-term learning.

4.4 Study of Interactive Reinforced Trainer

We study the impacts of proposed trainers for interactive reinforcement learning. We consider four variants: (1) *Non-trainer* removes both random forest trainer (RFT) and isolation forest trainer (IFT); (2) *RFT* removes IFT and takes only RFT for advice; (3) *IFT* removes RFT with only IFT for advice; (4) *RFT+IFT* has both two trainers for advice.

Figure 5 shows the efficiency comparison of each variant in terms of the explored best accuracy until current step. The result signifies that without trainers' help, agents need more steps for exploration to achieve better results, and both trainers can help for more efficient exploration and agent learning, especially when both the feature agent and the instance agent take external advice.

4.5 Case Study of Exploration Process

We also try to study and visualize the exploration process of agents. Figure 6 shows the accuracy in different steps with respect to selected feature number and instance number on FC dataset and Spam dataset. We visualize 10,000 points of exploration (colored in blue) and mark the top 10 best performed points in red. We easily observe that most blue points are on the certain part of data space, for example the space where feature number is (30 to 50) and instance number is (5,000 to 10,000) of the left subfigure. This signifies after the initial exploration, the exploration efficiently concentrates on the optimal data subspace. Then, the agents can continue searching until finally finding out the best feature and instance selection results.

5 Related Work

Feature selection includes three kinds of methods: (1) Filtering methods rank features based on relevance scores and select top-ranking ones (e.g., univariate feature selection). (2) Wrapper methods use predictors, considering the prediction performance as objective function (e.g., branch and bound algorithms). (3) Embedded methods incorporate feature selection into the classifier construction to search for an optimal feature subset (e.g., LASSO [Tibshirani, 1996]). **Instance selection** mainly includes two kinds of methods: (1) wrapper
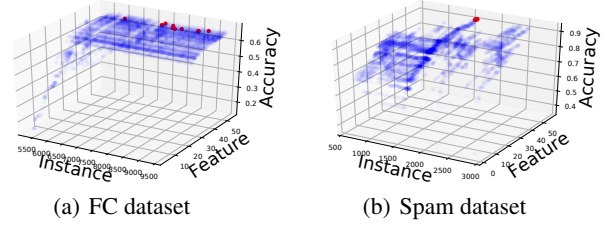
methods (e.g., *k-NN* based selection); (2) filter methods (e.g., $kd-$trees). The criterion is based on the accuracy obtained by a classifier. Most of wrapper methods are based on *k-NN* classifier. Other methods select instances by using SVM or Evolutionary algorithms, or are accomplished by finding border instances [Olvera-López *et al.*, 2010]. **Joint selection of Feature and Instance** has also been made some efforts to study; however, limited algorithms were proposed to tackle feature and instance selection simultaneously. The joint selection is firstly studied by genetic algorithms [Kuncheva and Jain, 1999]. Many researchers have tried to solve the joint selection task with heuristic search; they mainly are simulated annealing algorithms [De Souza *et al.*, 2008], cooperative coevolutionary algorithms [Derrac *et al.*, 2012]. Others explored the joint selection with principal component analysis [Suganthi and Karunakaran, 2019] or application of pairwise similarity [Benabdeslem *et al.*, 2020] or sequential forward search. Feature Selection and Instance Selection on Clinical Datasets uses Co-operative Co-evolution for classification [Olvera-López *et al.*, 2010]. **Reinforced Feature Selection** has applied reinforcement learning for the feature selection task. Some studies use single agent for feature selection [Fard *et al.*, 2013]. Also, multi-agent reinforcement learning have been used to automate feature selection [Liu *et al.*, 2019; Fan *et al.*, 2020]. Inspired by these work, we apply reinforcement learning into the joint selection task.

6 Conclusion

In this paper, we propose a dual-agent reinforcement learning framework towards the feature and instance joint selection task. We formulate the joint selection into a reinforcement learning framework with the tailor-designed sequential-scanning mechanism and collaboratively-changing environment, in order to simulate fine-grained interaction of feature space and instance space. The selection knowledge of the random forest trainer and the isolation forest trainer is applied to improve the efficiency of agent learning. The extensive experiments have demonstrated the superiority of our model on data preprocessing, which also reveals a workable design of reinforcement learning on the joint selection task.

Acknowledgements

References

[Aggarwal and Yu, 2001] Charu C Aggarwal and Philip S Yu. Outlier detection for high dimensional data. In *Proceedings of the 2001 ACM SIGMOD international conference on Management of data*, pages 37–46, 2001.

[Amir *et al.*, 2016] Ofra Amir, Ece Kamar, Andrey Kolobov, and Barbara Grosz. Interactive teaching strategies for agent training. In *25th International Joint Conference on Artificial Intelligence*, pages 2600–2607, 2016.

[Benabdeslem *et al.*, 2020] Khalid Benabdeslem, Dou El Kefel Mansouri, and Raywat Makkhongkaew. scos: Semi-supervised co-selection by a similarity preserving approach. *IEEE Transactions on Knowledge and Data Engineering*, 2020.

[Brighton and Mellish, 2002] Henry Brighton and Chris Mellish. Advances in instance selection for instance-based learning algorithms. *Data mining and knowledge discovery*, 6(2):153–172, 2002.

[Cai *et al.*, 2010] Deng Cai, Xiaofei He, Jiawei Han, and Thomas S Huang. Graph regularized nonnegative matrix factorization for data representation. *IEEE transactions on pattern analysis and machine intelligence*, 33(8):1548–1560, 2010.

[Chou *et al.*, 2006] Chien-Hsing Chou, Bo-Han Kuo, and Fu Chang. The generalized condensed nearest neighbor rule as a data reduction method. In *18th International Conference on Pattern Recognition (ICPR'06)*, volume 2, pages 556–559. IEEE, 2006.

[De Souza *et al.*, 2008] Jerffeson Teixeira De Souza, Rafael Augusto Ferreira Do Carmo, and Gustavo Augusto Lima De Campos. A novel approach for integrating feature and instance selection. In *2008 International Conference on Machine Learning and Cybernetics*, volume 1, pages 374–379. IEEE, 2008.

[Derrac *et al.*, 2010] Joaquín Derrac, Salvador García, and Francisco Herrera. Ifs-coco: Instance and feature selection based on cooperative coevolution with nearest neighbor rule. *Pattern Recognition*, 43(6):2082–2105, 2010.

[Derrac *et al.*, 2012] Joaquín Derrac, Isaac Triguero, Salvador García, and Francisco Herrera. Integrating instance selection, instance weighting, and feature weighting for nearest neighbor classifiers by coevolutionary algorithms. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 42(5):1383–1397, 2012.

[Dua and Graff, 2017] Dheeru Dua and Casey Graff. UCI ML repository. http://archive.ics.uci.edu, 2017.

[Fan *et al.*, 2020] Wei Fan, Kunpeng Liu, Hao Liu, Pengyang Wang, Yong Ge, and Yanjie Fu. Autofs: Automated feature selection via diversity-aware interactive reinforcement learning. In *2020 IEEE International Conference on Data Mining (ICDM)*, pages 1008–1013. IEEE, 2020.

[Fard *et al.*, 2013] Seyed Mehdi Hazrati Fard, Ali Hamzeh, and Sattar Hashemi. Using reinforcement learning to find an optimal set of features. *Computers & Mathematics with Applications*, 66(10):1892–1904, 2013.

[Guo *et al.*, 2018] Minghao Guo, Jiwen Lu, and Jie Zhou. Dual-agent deep reinforcement learning for deformable face tracking. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 768–783, 2018.

[Kuncheva and Jain, 1999] Ludmila I Kuncheva and Lakhmi C Jain. Nearest neighbor classifier: Simultaneous editing and feature selection. *Pattern recognition letters*, 20(11-13):1149–1156, 1999.

[Liu and Motoda, 2012] Huan Liu and Hiroshi Motoda. *Feature selection for knowledge discovery and data mining*, volume 454. Springer Science & Business Media, 2012.

[Liu *et al.*, 2008] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. Isolation forest. In *2008 Eighth IEEE International Conference on Data Mining*, pages 413–422. IEEE, 2008.

[Liu *et al.*, 2019] Kunpeng Liu, Yanjie Fu, Pengfei Wang, Le Wu, Rui Bo, and Xiaolin Li. Automating feature subspace exploration via multi-agent reinforcement learning. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 207–215, 2019.

[Lu and Weng, 2007] Dengsheng Lu and Qihao Weng. A survey of image classification methods and techniques for improving classification performance. *International journal of Remote sensing*, 28(5):823–870, 2007.

[Mnih *et al.*, 2013] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

[Olvera-López *et al.*, 2010] J Arturo Olvera-López, J Ariel Carrasco-Ochoa, J Francisco Martínez-Trinidad, and Josef Kittler. A review of instance selection methods. *Artificial Intelligence Review*, 34(2):133–143, 2010.

[Pal, 2005] Mahesh Pal. Random forest classifier for remote sensing classification. *International journal of remote sensing*, 26(1):217–222, 2005.

[Suganthi and Karunakaran, 2019] M Suganthi and V Karunakaran. Instance selection and feature extraction using cuttlefish optimization algorithm and principal component analysis using decision tree. *Cluster Computing*, 22(1):89–101, 2019.

[Sutton and Barto, 2018] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[Tibshirani, 1996] Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1):267–288, 1996.

[Tsai *et al.*, 2013] Chih-Fong Tsai, William Eberle, and Chi-Yuan Chu. Genetic algorithms in feature and instance selection. *Knowledge-Based Systems*, 39:240–247, 2013.

[Wilson and Martinez, 2000] D Randall Wilson and Tony R Martinez. Reduction techniques for instance-based learning algorithms. *Machine learning*, 38(3):257–286, 2000.