Fast and Adaptive Multi-Objective Feature Selection for Classification

1st Fei Ming[1,2], 2nd Bing Xue[2], 3th Mengjie Zhang[2], and 4th Yaochu Jin[1]
[1] *School of Engineering*, *Westlake University*, Hangzhou 310030, China. Emails: {mingfei;jinyaochu}@westlake.edu.cn
[2] *Center of Data Science and Artificial Intelligence & School of Engineering and Computer Science*
Victoria University of Wellington, Wellington 6140, New Zealand. Emails: {bing.xue;mengjie.zhang}@ecs.vuw.ac.nz

***Abstract*—Identifying high-quality feature subsets for decision-makers by wrapper-based multi-objective feature selection (MOFS) has been attracting increasing attention. As a mainstream approach, evolutionary methods offer distinct advantages but also struggle with increasingly complex application scenarios and high-dimensional data, mainly in terms of time efficiency, exponentially expanding search space, and adaptive determination of a suitable classifier. To overcome these challenges, this paper proposes two simple yet effective methods: Fast Initialization (FI) and one-generation Adaptive K-Nearest Neighbor (AK). FI leverages mutual information and tournament selection to locate high-quality initial feature subsets computationally efficiently. AK verifies that, with theoretical proof, using a single generation can determine the most suitable KNN for different data to improve feature selection performance with very little time overhead and without any data analysis or assumption. Experiments on 20 real-world high-dimensional datasets demonstrate the superior performance of FI and AK to advanced initialization and KNN methods for MOFS. We also validated that the obtained feature subsets generalize well to an LLM for tabular data, enabling it to be seamlessly applied to high-dimensional data and achieve superior performance.**

***Index Terms*—Multi-objective feature selection, initialization, adaptive KNN, classification, large language model.**

I. Introduction

In the era of increasing informatization and intelligence across various industries, classification tasks in real-world often encounter great challenges such as high-dimensional data [1], [2], complex sample distributions (*e.g.,* imbalance and outliers) [3], and the lack of interpretability in deep learning models [4]. Feature selection (FS), as a data preprocessing technique, aims to identify small yet informative feature subsets. This not only enhances the interpretability of classification but also reduces training and inference costs, lowers data collection expenses, removes unimportant and noisy features, and mitigates overfitting to improve classification performance [5].

In particular, multi-objective feature selection (MOFS) has attracted growing attention in recent years [6], as it enables the discovery of a set of Pareto-optimal feature subsets, named Pareto front (PF), that represent trade-offs between classification performance and feature subset size. These trade-off solutions offer valuable flexibility for different decision-makers and application scenarios.

Wrapper-based FS using evolutionary computation (EC) methods has become a mainstream approach to MOFS due to the following reasons. First, real-world data often involve complex feature interactions. EC-based wrapper methods can capture and retain these interactions during the search to improve classification performance [7]. Second, as a population-based search approach, EC is inherently well-suited for MOFS, offering both global search ability free from problem's analytical expression or gradient availability and the generation of diverse Pareto-optimal solutions in a single run [8].

Despite the advantages, it is still very challenging to handle high-dimensional data due to the following issues. First, evolutionary iterations and classification performance evaluation results in substantial time overhead [9]. Second, the limited number of evaluations greatly increases the difficulty of identifying high-quality subsets in high-dimensional feature spaces. Third, as a lazy model requiring no complex training process at each iteration and is easy to implement with high performance, k-nearest neighbour (KNN) [10] is widely used as the classifier for wrapper-based feature selection in classification. However, real-world high-dimensional data exhibit complex and diverse characteristics, such as class imbalance and distributions of data points. Thus, using a fixed KNN with a deterministic k value and ad-hoc neighbour voting mechanism is hard to handle complex data. Therefore, it is desired to address these challenges, improve the performance, and reduce the time consumption of evolutionary MOFS methods to better handle complex real-world high-dimensional data.

The initialization strategy plays a vital role in addressing the second issue since a proper initial population can facilitate the convergence in a high-dimensional search space and avoid getting trapped in local optima. However, existing initialization techniques in MOFS methods either fail to identify high-quality features through training data analysis, thus resulting in excessive randomness [11], or features are selected based solely on their importance with poor population diversity risking local optima [1], [12], or the process is too time-consuming [13], [14]. For the third issue, considering the high computational cost of wrapper-based methods and the coherence between feature subset quality and the adopted classifier, it is desired yet challenging to automatically determine the most suitable KNN settings (k value and neighbor voting

This work was supported by the International Collaboration Fund for Creative Research of National Science Foundation of China (NSFC ICFCRT) under Grant W2441019 and the National Natural Science Foundation of China under Grant 62136003.(Corresponding author: Yaochu Jin)

mechanism) for the given dataset, but without introducing much overhead. Although adaptive KNN selection or learning for classification has been well-studies [15], [16], existing methods are not tailored for wrapper-based MOFS and are thus inapplicable unless embedded into each iteration, which will inevitably cause unacceptable time consumption.

This paper addresses these issues and improves the effectiveness and efficiency of MOFS from both feature and classifier aspects[1]. The main contributions are as follows:

1). A **Fast Initialization (FI)** is proposed that measures the mutual information (MI) [17] between each feature and class label as feature importance, and uses a parameter-adaptive tournament selection to balance feature importance and population diversity. FI achieves competitive accuracy compared to existing methods and is much more efficient.

2). An **Adaptive KNN (AK)** is developed to automatically determine the most suitable KNN for the given dataset with little overhead and without any assumption. Multiple independent populations evolve in the first generation, each using a KNN to preserve the coherence between features' quality and the classifier. Then, the population and its KNN with the best classification performance are selected to finish evolution.

3). The new method integrating FI and AK outperforms representative and advanced methods in experiments on 20 real-world high-dimensional datasets. Moreover, the obtained feature subsets generalize well to an LLM tailored for tabular data, which cannot be directly applied on these datasets due to high dimensionality. However, our feature selection method enables it to be directly applied to high-dimensional data and rapidly achieve superior performance.

II. Related Work

A. Initialization Methods in MOFS

Existing initialization methods, based on whether the information of the data is used, can be divided into the following categories. A detailed review is presented in the Supplementary file.

Some methods propose to use the information extracted from the data to locate important features [13], [14], [18]–[20]. These methods use training data information to assist initialization, which might benefit high-dimensional MOFS. However, most of them introduce unacceptable overhead. Assuming the data contains d features and N samples, then the time consumption of calculating the correlation measure between each pair of features can be $O(d^2N)$. Therefore, a time-efficient method is desired.

Quite a few evolutionary MOFS methods randomly generate the initial population [11], [21], [22], which can significantly reduce time consumption but is ineffective in handling high-dimensional data. Others use population information to guide initialization [1], [12], [23]. In summary, random initialization has much lower time cost ($O(dN_p)$ where N_p is the population size) and population information-based initialization can also be time-consuming (maximal $O(d^2N_p)$ but usually after feature selection, d is significantly reduced); however, due to the exponentially growing search space of high-dimensional problems, it is challenging to identify valuable features from a large number of features using limited individuals without utilizing the information from the training data.

[1]In this work, the KNN selection aims to select a KNN for the whole dataset, rather than for each sample.

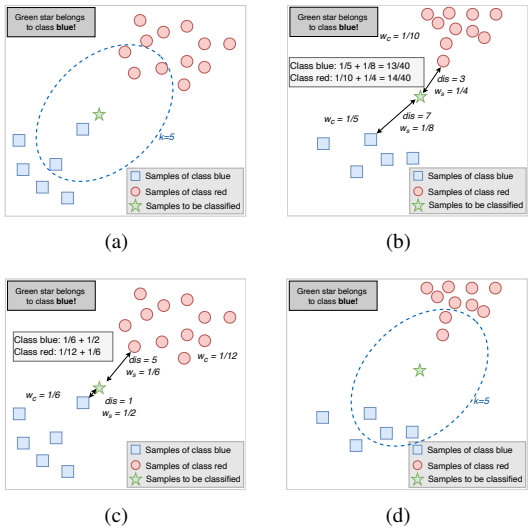


Fig. 1. Limitations of using a fixed KNN on different data. (a). Using KNN with k set to 5, where the classification result is incorrect; (b). Under the same situation as (a), KNN2W can correctly classify; (c). A situation where KNN2W obtains an incorrect classification result due to the incomplete pattern reflected by the insufficient samples of the minority class; (d). Under the same situation as (c), KNN can correctly classify. Such phenomena indicate that a fixed KNN in the MOFS algorithm is hard to handle distinct datasets.

B. KNNs in MOFS for Classification

Owing to the promising classification performance and easy-to-implement nature, KNN is widely used as the classifier in classification [16]. The common practice of using KNN in MOFS for classification includes two paths. One is using KNN with a fixed k value [6], [23], and another is improving the original KNN by a weighted approach such as the double-weighted KNN (named KNN2W) proposed in [1]. However, as the dimensionality of the data increases and the distribution becomes complex, using a pre-defined and fixed KNN is less effective for different data. These limitations on complex data are illustrated by an artificial scene shown in Fig. 1 and analyzed in the Supplementary file.

In summary, the above situations show that when the feature dimensionality increases and the sample distribution becomes complex, using a single pre-defined KNN can be ineffective for different data.

III. Proposed Methods

The MOFS problem in this work can be formulated as:

$$\begin{aligned} \text{Minimize} \quad & \mathbf{F}(\mathbf{x}) = (f_1(\mathbf{x}), f_2(\mathbf{x}))^T \\ \text{subject to} \quad & \mathbf{x} \in \mathbf{\Omega}^d \end{aligned}. \tag{1}$$

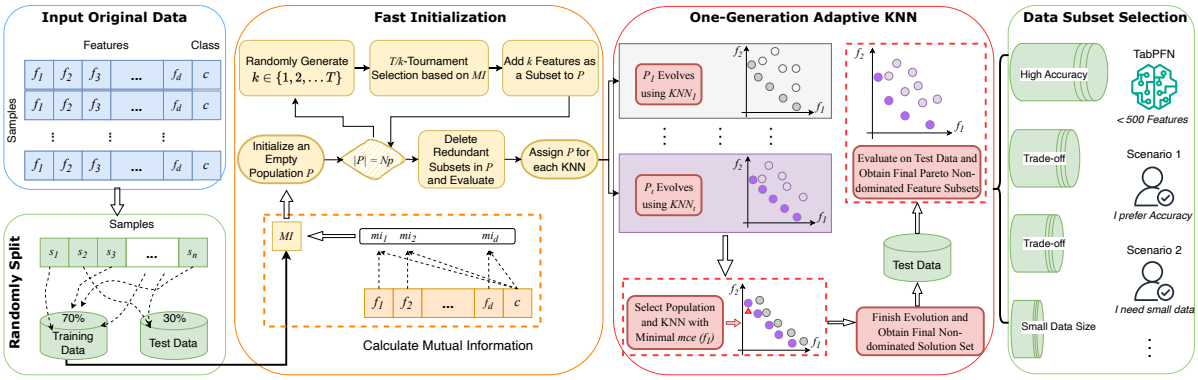


Fig. 2. Overall flowchart of the **H**igh-dimensional **MOFS** framework based on our proposed **FI** and **AK** (HMOFS-FI-AK).

The balanced classification error f_1 is

$$f_1(\mathbf{x}) = 1 - \frac{1}{c}\sum_{i=1}^{c}\frac{TP_i}{S_i}, \tag{2}$$

where c is the number of classes, TP_i is the true positive samples in the i-th class, and $|S_i|$ is the number of samples in the i-th class. The weight for each class is set to $1/c$ to avoid biases to the majority classes for imbalanced data.

f_2 is the selected feature ratio calculated by

$$f_2(\mathbf{x}) = \frac{\sum_{i=1}^{d} x_i}{d}, \tag{3}$$

where $\mathbf{x} = (x_1, \dots, x_d)^T$ is the decision vector with d decision variables (*i.e.,* features); $x_i = 1$ means the i-th feature is selected, and $x_i = 0$ means not; $\mathbf{x} \in \mathbf{\Omega}^d$ is the search space (decision space), and $\mathbf{F} \in \mathbf{\Omega}^m$ is the objective space.

A. Framework

The overall flowchart of our proposed high-dimensional MOFS framework based on FI and AK (HMOFS-FI-AK) is presented in Fig. 2. For a dataset with N samples, each having d features and one class label, we randomly split it into a training set (70%) and a test set (30%). Then, the MI is calculated based on the training set, and the proposed FI is conducted. Afterward, the initialized population is assigned to t KNNs for evolution of one generation. Afterward, the KNN and population with the best minimal classification error is selected to finish the evolution. We then obtain the non-dominated solutions from the final population and evaluate them on the test set to select only the non-dominated feature subsets as the final output. So, HMOFS-FI-AK can obtain Pareto feature subsets for different practitioners and scenarios. For example, the LLM **TabPFN** [24] tailored for tabular data strictly requires no more than 500 features.

B. Fast Initialization

The pseudocode of FI is summarized in Algorithm 1. The MI between the j-th feature X_j and the class label Y is calculated by:

$$I(X_j;Y) = \sum_{x\in\mathcal{X}_j}\sum_{y\in\mathcal{Y}}\frac{n_{x,y}}{N}\log\left(\frac{n_{x,y}\cdot N}{n_x \cdot n_y}\right), \tag{4}$$

Algorithm 1 Fast Initialization

Input: $\mathcal{D}$ (Training Data), N (# Samples), d (# Features), N_p (Population Size), T (Sparse Factor)
Output: $\mathcal{P}$ (Initialized Population)

1: $MI \leftarrow$ Create an empty list with size d;
2: **for** $col = 1$ **to** d **do**
3: $MI(col) \leftarrow$ Calculate MI of col-th feature to class label by Equation (4);
4: **end for**
5: $\mathcal{P} \leftarrow$ Initialize the population as an $N_p \times d$ all-zero matrix;
6: **for** $i = 1$ **to** N_p **do**
7: Generate a random integer k, $k \in \{1, 2, \cdots, T\}$;
8: $\mathbf{S} \leftarrow$ Initialize selected feature list as an empty list;
9: **for** $j = 1$ **to** $\lceil T/k \rceil$ **do**
10: $C \leftarrow$ Randomly choose k distinct indices from $\{1, 2, \cdots, d\}$;
11: $best \leftarrow \arg\max_{l\in C} MI(l)$;
12: Append $best$ to $\mathbf{S}$;
13: **end for**
14: $\mathcal{P}(i, \mathbf{S}) \leftarrow 1$;
15: **end for**
16: $\mathcal{P} \leftarrow$ Delete redundant feature subsets;
17: **return** $\mathcal{P}$.

where X_j is the j-th feature column, for $j = 1, 2, \dots, d$; Y is the class label column; $\mathcal{X}_j$ is the set of unique values taken by feature X_j; $\mathcal{Y}$ is the set of all possible class labels; $n_{x,y}$ is the number of samples where $X_j = x$ and $Y = y$; n_x is the number of samples where $X_j = x$; n_y is the number of samples where $Y = y$; N is the total number of samples.

The probabilities $p(x)$, $p(y)$, and $p(x, y)$ are estimated empirically from the dataset using frequency counts, *i.e.,*

$$p(x,y) \approx \frac{n_{x,y}}{N}, \quad p(x) \approx \frac{n_x}{N}, \quad p(y) \approx \frac{n_y}{N}. \tag{5}$$

For continuous features, we simply discretize them using the equal-frequency with 10 bins.

Since we don't calculate the MI between features, the time consumption is much less than existing MI-based methods. Moreover, a parameter-adaptive mechanism is designed (lines 7-14), which leverages the MI and tournament selection to balance feature importance and population diversity. Specifically, a large k leads to a small feature subset ($\lceil T/k \rceil$). Therefore, we select more important features by a k-tournament that selects the feature with the largest MI among k randomly selected fea-

Algorithm 2 MOFS Framework based on Adaptive KNN

Input: $MOFS$ (Adopted MOFS Algorithm for Evolution), $\{KNN_1, \cdots, KNN_t\}$ (KNNs)
Output: $\mathcal{P}$ (Final Population, *i.e.,* Feature Subsets)
1: $\mathcal{P} \leftarrow$ Initialize a population using FI by Algorithm 1;
2: Assign $\mathcal{P}$ to populations $\mathcal{P}_1$ to $\mathcal{P}_t$ and correspondingly use each KNN: KNN_1 to KNN_t as classifier to evaluate;
3: $\mathcal{P}_1$ to $\mathcal{P}_t$ evolve one generation independently by $MOFS$ algorithm and the corresponding KNN;
4: $\mathcal{P}, KNN \leftarrow$ Determine the population with the lowest minimal classification error among all individuals, and get its KNN;
5: $\mathcal{P} \leftarrow$ Finish the evolution starting with $\mathcal{P}$ and using KNN by $MOFS$;
6: $\mathcal{P} \leftarrow$ Evaluate on test data and select non-dominated solutions as the final PF;
7: **return** $\mathcal{P}$ as PF.

tures. On the contrary, a small k leads to a large feature subset. Then, we select more diverse features by k-tournament from a few randomly selected features. Consequently, small feature subsets contain important features to improve classification performance, while large feature subsets provide diversity for exploring the search space.

C. Adaptive KNN

The pseudocode of the proposed AK is summarized in Algorithm 2. While wrapper-based EC methods are effective, their high computational cost renders the use of classifier ensembles often infeasible in practical scenarios. However, in AK, all populations only evolve one generation using their corresponding KNNs. Afterward, the most suitable KNN is determined according to the minimal f_1 value (classification error). Therefore, this simple mechanism introduces only little overhead, requires no data analysis or assumption, and is easy to extend or implement.

The underlying principles and theoretical analysis of AK are as follows. Detailed proof is presented in the Supplementary file. Let $\mathcal{P}^{(t)}$ denote the population of feature subsets in generation t; meanwhile, let $p = 1/d$ be the per-bit mutation rate on a d-dimensional feature string. We model the population sequence $\{\mathcal{P}^{(t)}\}$ as a finite–state Markov chain [25]. The mutation step flips a binomial number of bits whose tail probability is exponentially small by the Chernoff bound [26], [27], so offspring remain within an $O(\ln d)$ Hamming ball of their parents with high probability. Moreover, the expected progress of the best individual can be bounded using the locality of evolutionary operators and drift analysis [28], [29], which characterizes the expected one–step decrease of a distance-to-optimum potential function and implies that the search process concentrates around the local optimum at a rate governed by the mutation strength. With classifier accuracy assumed L–Lipschitz in Hamming distance and the initial population covering a δ-radius neighborhood of the local optimum with fraction ρ (i.e., at least a ρ proportion of individuals lie within Hamming distance δ of the local maximizer), the empirical accuracy gap Δ between the best configuration and all others cannot shrink by more than $O(\delta)$ per generation. If the initial margin satisfies $\rho\Delta > 4\rho L\delta$, a union bound over G generations (each generation violating the locality or drift condition with probability at most η, where η bounds the chance that mutation or crossover escapes the δ-ball) guarantees that the configuration achieving the largest initial accuracy remains the unique maximizer with probability at least $1 - \big(G\eta + \xi\big)$. This establishes one–generation identifiability and multi–generation stability of the selected KNN configuration.

IV. Experimental Settings

In this paper, we embed FI and AK into the DAEA [12] algorithm. Then, we compare FI with random initialization and the initialization methods of DAEA [12], SMII [14], JSEMO [1], and FPPFS [13], and compare AK to KNN ($k = 5$) and KNN2W (denoted as JSCla). Experiments were conducted on PlatEMO [30] and TabPFN [24] using a local computer with Intel Ultra9 285K CPU, 32 GB RAM, and RTX 5080 16G in Windows 11 operating system.

Datasets: We adopt 20 real-world high-dimensional datasets whose detailed information is presented in the Supplementary file. The datasets are all open-source data. Due to the large dimensions of most data sets, the experiments were extremely time-consuming for [13], [14], so only several datasets are used in these two algorithms.

Parameter Settings of the EA: Following most MOFS studies, the evolutionary parameters are set as follows. The population size N_p is 200. The maximal number of individual evaluations E_{max} is 20000, and the maximal generations G_{max} is $\lceil E_{max}/N_p \rceil$. In our methods, FI contains a parameter T, which is set to 400. All parameters of other algorithms and methods are the same as in their original literature.

Performance Indicators: The hypervolume (HV) [31] is adopted to measure the performance in the multi-objective optimization aspect. Since the true PFs are unknown, we save the final results of all algorithms. Then, we extract the non-dominated solutions and use their objective vectors to get the ideal and nadir points for normalization. After normalization, we use (1.1,...,1.1) as the reference point for HV calculation. In addition, we also record the minimum classification error (MCE) of the final population of each algorithm as the classification performance.

Statistical Analysis: Each algorithm independently executes 30 times on each dataset. The mean and standard deviation values of HV and MCE are recorded. The Wilcoxon rank-sum test with a significance level of 0.05 is employed. "+", "−", and "≈" indicate the result of another algorithm is significantly better than, significantly worse than, and statistically similar to our methods. The Friedman test with a Holm correction at a significance level of 0.05 is also used to obtain rankings and p-values of different algorithms.

V. Results and Analyses

A. FI and Existing Initialization Methods

The results of FI compared to existing methods in MOFS literature and random initialization are summarized in Table I. FI achieves better HV and MCE results than all other methods, showcasing a significant advantage in HV. The HV advantage means HMOFS-FI finally obtains a Pareto feature subset

with significantly better convergence (minimization of both classification error and subset size) and diversity (more Pareto solutions). This indicates that FI can provide a high-quality initial population to facilitate the convergence and diversity of evolutionary search. Moreover, on CPU run time, FI also achieves promising results. Although DAEA performs better on run time, it performs significantly worse on HV and slightly worse on MCE.

Fig. 3 depicts the mean values of CPU run time of each method on each dataset. From the results, FI is much more time-efficient than other methods, especially on datasets with very high dimensionality. Among the peer initialization methods, JSEMO can be time-consuming, while DAEA is time-efficient. Nevertheless, the initialization methods of JSEMO and DAEA both perform worse.

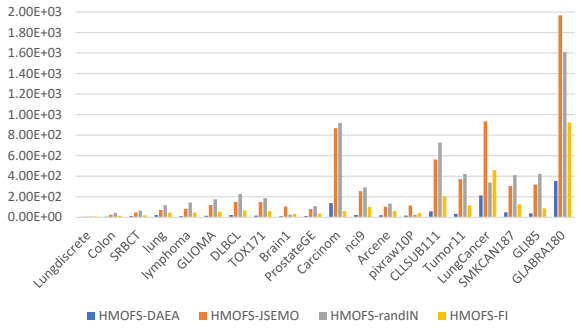


Fig. 3. CPU run time of using different initialization methods on all datasets.

In this part, we also conduct comparisons including SMII and FPPFS on several relatively low-dimensional datasets. Detailed results and analysis are presented in the Supplementary file. It can be observed that the time overhead of SMII and FPPFS increases sharply with the dimensionality, reaching an unacceptable level. In contrast, DAEA and FI are less sensitive to dimensionality changes. However, the initialization strategy of DAEA does not consider information from the data, resulting in inferior performance compared to FI in terms of HV and MCE.

In summary, FI can provide a high-quality initial population for evolutionary MOFS search in a time-efficient way. Detailed results are presented in the Supplementary file.

In the above experiments, SMII [14] and FPPFS [13] are not conducted on all datasets because they are extremely time-consuming on high-dimensional datasets, which hinders their feasibility in many real-world applications. In this part, we conduct comparisons including SMII and FPPFS on several relatively low-dimensional datasets. The CPU run time results

TABLE I
RESULTS OF FI AND OTHER INITIALIZATION METHODS, INCLUDING THE FRIEDMAN RANKINGS AND WILCOXON TEST RESULTS.

	HV		MCE		Time	
	Ranking	+/-/=	Ranking	+/-/=	Ranking	+/-/=
HMOFS-DAEA [12]	2.225	0/12/8	2.35	0/1/19	**1.05**	19/1/0
HMOFS-JSEMO [1]	2.5	0/10/10	2.175	0/4/16	3.15	1/19/0
HMOFS-Random	3.575	0/17/3	3.5	0/13/7	3.6	4/16/0
HMOFS-FI	**1.8**		**1.925**		2.2	

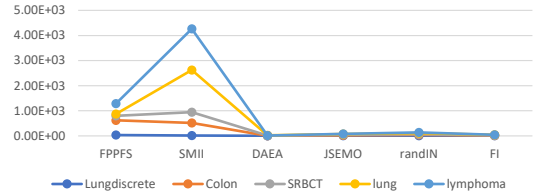
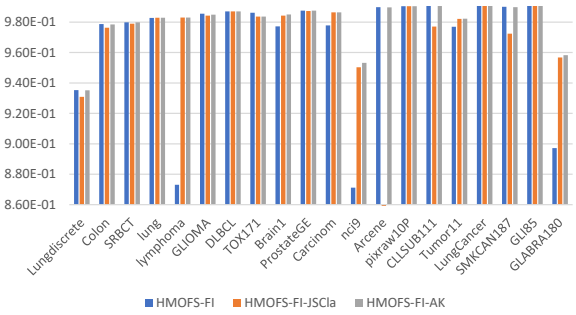


Fig. 4. CPU run time of using different initialization methods on the first five datasets.

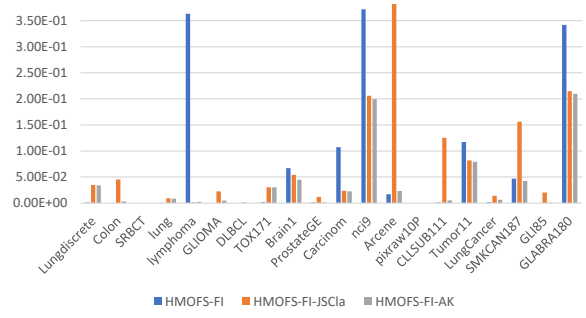
are presented in Fig. 4. It can be observed that the time overhead of SMII and FPPFS increases sharply with the dimensionality, reaching an unacceptable level. In contrast, DAEA and FI are less sensitive to dimensionality changes. However, the initialization strategy of DAEA does not consider information from the data, resulting in inferior performance compared to FI in terms of HV and MCE.

B. AK and Single Fixed KNN

The HV and MCE results of using KNN ($k = 5$) or KNN2W [1] and AK to adaptively select from them are presented in Fig. 5. Although HMOFS-FI-AK does not achieve the best performance on every dataset, the HV comparisons on each dataset show that HMOFS-FI-AK consistently outperforms the worse one between KNN ($k = 5$) and KNN2W, indicating that AK is capable of selecting the more suitable KNN. The slight performance degeneration is attributed to the fact that different KNNs consume part of the very limited evaluation budget in the first generation.



(a)



(b)

Fig. 5. HV (a) and MCE (b) of using KNN ($k = 5$), KNN2W, and our proposed AK method.

The difference is more pronounced in terms of MCE: when the more suitable KNN is selected for a given dataset,

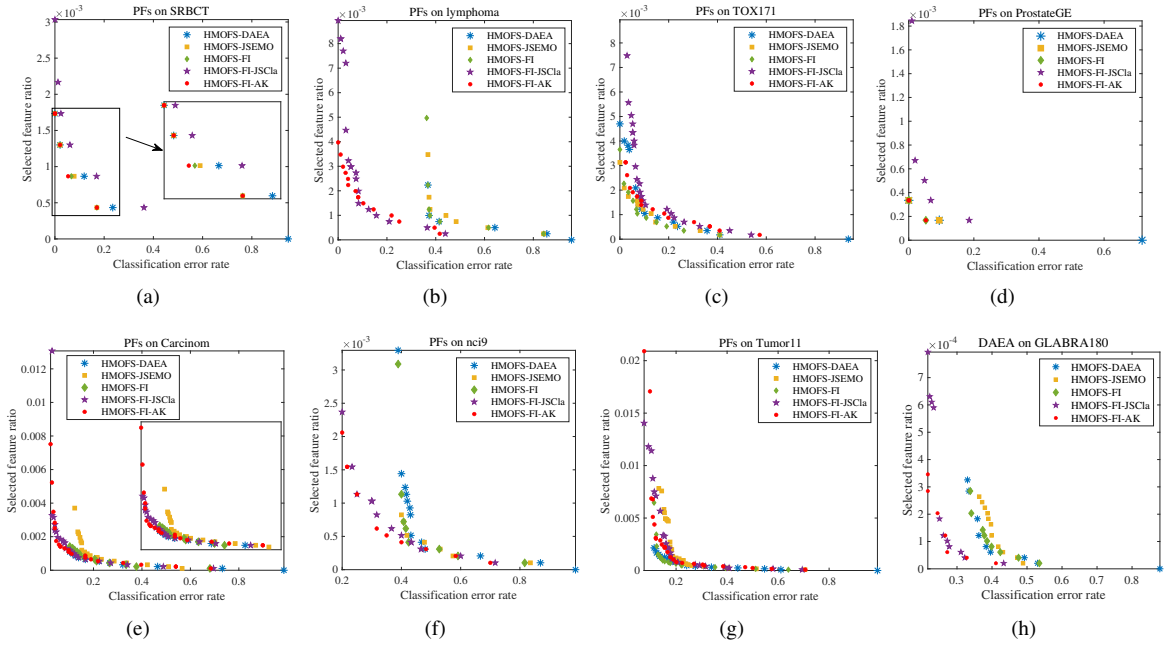


Fig. 6. PFs obtained by different methods on test sets of representative datasets with the median HV values among the 30 runs.

the final MCE can be significantly reduced. Similarly, AK consistently achieves better results than the worst-performing KNN, indicating its ability to select the more appropriate KNN. The slightly inferior results are due to the same reason mentioned above.

In summary, AK can adaptively determine the most suitable KNN based on the experimental results. Detailed results are presented in the Supplementary file.

C. Comparisons on PFs and Convergence Profiles

To intuitively compare the final PFs, we select eight representative datasets and plot the final PFs using different initialization methods and KNNs in Fig. 6.

SRBCT is of relatively low dimensionality, so the performance of HMOFS-FI-AK is just slightly better than HMOFS-FI. This reveals that FI performs better and KNN ($k = 5$) is more suitable. Moreover, it also verifies that AK selects the correct KNN, so it performs much better than HMOFS-FI-JSCla. On lymphoma, it can be found that KNN2W is much more effective, AK can select KNN2W, and FI can help improve the convergence. On TOX171, HMOFS-FI and HMOFS-JSEMO perform well, and HMOFS-FI-AK can determine KNN ($k = 5$). On ProstateGE, HMOFS-FI and HMOFS-FI-AK achieve the best performance, revealing that FI is effective and KNN ($k = 5$) is determined by AK. On Carcinom and nci9, HMOFS-FI-AK obtains the best performance with KNN2W selected. On Tumor11, using KNN ($k = 5$) achieves better overall convergence but KNN2W brings better classification performance. AK somehow achieves a trade-off between them. On GLABRA180, HMOFS-FI-AK obtains the best PF, where FI performs similar to DAEA, but AK significantly improves the final performance. [2]

The convergence profiles of HV and MCE on five representative datasets are plotted in Fig. 7. On Brain1, nci9, Tumor11, and GLABRA180, KNN2W is more suitable, and AK can determine this at the first generation and then quickly catch up with its performance and eventually achieve comparable or even better results. On CLLSUB111, KNN ($k = 5$) is more suitable and AK can also detect. On all datasets, the header start and better convergence performance verify the effectiveness of FI.

D. Applying to LLM TabPFN

To verify that our MOFS results can generalize well to popular deep learning models, we obtain the final feature subset of each algorithm with the best balanced accuracy, including the original DAEA and JSEMO algorithms. Then, we use these features to form data subsets. Afterward, these data subsets can be directly used in LLM **TabPFN**. It should be noted that LLM **TabPFN** [24] cannot be directly used on these datasets due to the high dimensionality. TabPFN provides different measures: ROC AUC, balanced accuracy, and accuracy. The results are presented in Table II. On most datasets, FI-AK achieves the best results as FI provides a better initial population and AK determines a suitable KNN; on Carcinom and SMK-CAN-187, FI or FI-JSCla obtains the best, while FI-AK ranks second. These results demonstrate that FI is effective and AK can determine the most suitable KNN to assist in searching for high-quality feature subsets that generalize well to other models. It is also worth noting that

[2]Comparisons on convergence profiles are in the Supplementary file.

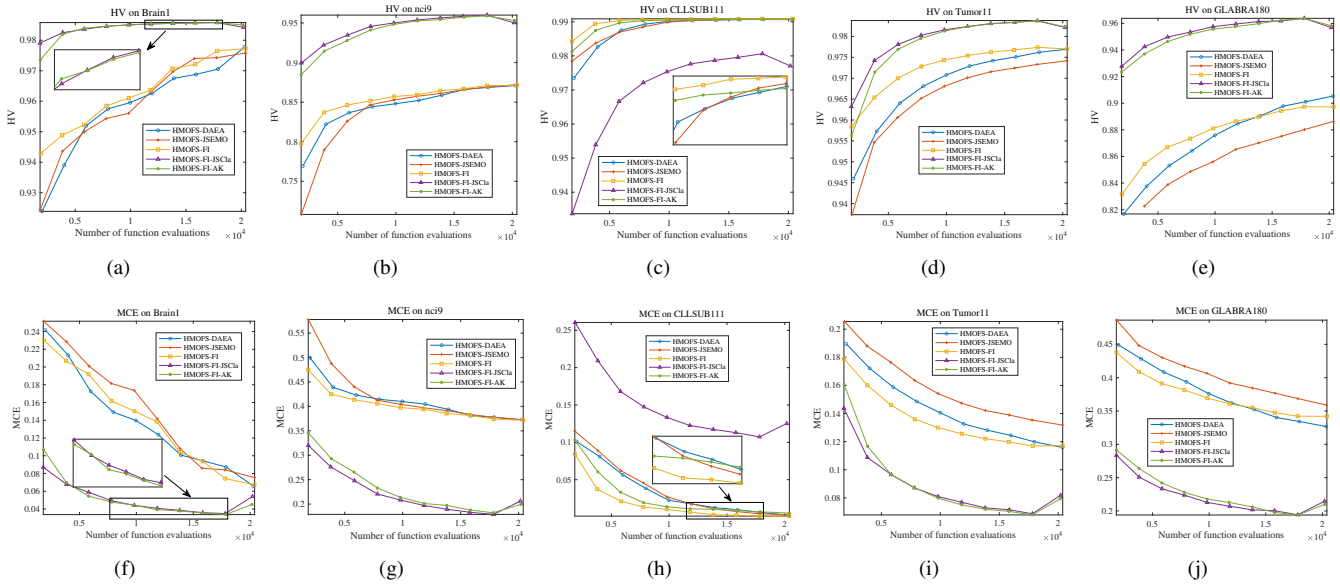


Fig. 7. Convergence profiles of HV and MCE indicators of different methods on five representative datasets with the median indicator values among the 30 runs. The final step shows the performance changes from the training to the test set.

TABLE II
ROC AUC, BALANCED ACCURACY, AND ACCURACY RESULTS OF IMPORTANT PEER METHODS AND OUR PROPOSED METHODS USING THEIR OBTAINED FEATURE SUBSETS FOR TABPFN.

Dataset	Measure	JSEMO	DAEA	FI	FI-JSCla	FI-AK	Dataset	Measure	JSEMO	DAEA	FI	FI-JSCla	FI-AK
lung_discrete	ROC AUC	**0.99**	**0.99**	0.97	**0.99**	**0.99**	Carcinom	ROC AUC	0.99	0.96	0.99	**1.00**	0.99
	Balanced Acc	0.94	0.89	0.67	**0.96**	**0.96**		Balanced Acc	0.86	0.83	0.83	**0.97**	0.87
	Accuracy	0.92	0.92	0.72	**0.96**	**0.96**		Accuracy	0.91	0.87	0.89	**0.96**	0.93
Colon	Balanced Acc	0.89	0.85	**0.96**	0.82	0.89	nci9	Balanced Acc	0.50	0.27	0.54	**0.63**	**0.63**
	Accuracy	0.90	0.85	**0.95**	0.85	0.90		Accuracy	0.50	0.25	0.55	**0.60**	**0.60**
SRBCT	ROC AUC	**1.00**	**1.00**	0.99	**1.00**	**1.00**	arcene	Balanced Acc	0.56	0.89	0.89	0.50	**0.93**
	Balanced Acc	0.95	0.95	0.85	**1.00**	**1.00**		Accuracy	0.56	0.89	0.89	0.56	**0.93**
	Accuracy	0.96	0.96	0.86	**1.00**	**1.00**	pixraw10P	ROC AUC	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
lung	ROC AUC	0.98	**0.99**	**0.99**	**0.99**	**0.99**		Balanced Acc	0.97	**1.00**	**1.00**	0.97	**1.00**
	Balanced Acc	0.89	0.96	0.89	**0.99**	0.96		Accuracy	0.96	**1.00**	**1.00**	0.96	**1.00**
	Accuracy	0.89	0.95	0.92	**0.97**	0.96	CLL_SUB_111	ROC AUC	0.89	0.86	0.87	0.83	**0.90**
lymphoma	ROC AUC	0.95	0.88	0.92	**0.99**	0.97		Balanced Acc	0.83	0.83	0.73	0.81	**0.86**
	Balanced Acc	0.81	0.78	0.77	0.81	**0.85**		Accuracy	0.79	0.78	0.64	0.75	**0.81**
	Accuracy	0.90	0.87	0.87	0.90	**0.94**	Tumor11	ROC AUC	0.98	0.95	**0.99**	**0.99**	**0.99**
GLIOMA	ROC AUC	0.91	0.91	0.84	0.96	**0.97**		Balanced Acc	0.84	0.75	0.83	0.77	**0.87**
	Balanced Acc	0.65	0.62	0.50	**0.95**	0.82		Accuracy	0.86	0.79	0.87	0.82	**0.89**
	Accuracy	0.76	0.64	0.58	**0.94**	0.88	Lung_Cancer	ROC AUC	**0.99**	0.98	0.95	**0.99**	**0.99**
DLBCL	Balanced Acc	0.69	0.75	**1.00**	0.97	**1.00**		Balanced Acc	**0.94**	0.87	0.90	0.89	**0.94**
	Accuracy	0.70	0.75	**1.00**	0.96	**1.00**		Accuracy	**0.97**	0.94	0.95	0.94	**0.97**
TOX-171	ROC AUC	0.95	0.94	0.98	0.95	**0.99**	SMK-CAN-187	Balanced Acc	0.73	0.70	**0.80**	0.72	0.77
	Balanced Acc	0.77	0.82	0.82	0.75	**0.88**		Accuracy	0.74	0.70	**0.80**	0.72	0.77
	Accuracy	0.77	0.82	0.82	0.75	**0.88**	GLI_85	Balanced Acc	0.86	0.80	0.81	0.86	**0.94**
Brain1	ROC AUC	0.96	0.85	0.95	**0.97**	0.95		Accuracy	0.89	0.86	0.82	0.89	**0.97**
	Balanced Acc	0.80	0.73	0.53	0.88	**0.89**	GLA_BRA_180	ROC AUC	0.88	0.86	0.86	**0.89**	**0.89**
	Accuracy	**0.93**	0.90	0.80	0.90	**0.93**		Balanced Acc	0.65	0.64	0.64	0.64	**0.79**
Prostate-GE	Balanced Acc	0.88	0.94	**0.97**	**0.97**	**0.97**		Accuracy	0.73	0.70	0.70	0.71	**0.81**
	Accuracy	0.88	0.94	**0.97**	**0.97**	**0.97**							

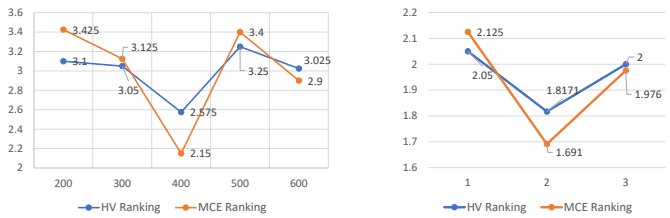
after feature selection, TabPFN can finish its inference in a few seconds on all these datasets. Therefore, it also demonstrates that feature selection is promising in significantly reducing the time-consumption of LLMs in fine-tuning and inference.

E. Sensitivity Analysis

In FI, T is used to control the sparsity, so its sensitivity is analyzed. In AK, we also conduct parameter studies using zero or two generations. Friedman test is performed on the HV and MCE results and summarized in Fig. 8. $T = 400$ in FI performs significantly better, and one generation in AK performs significantly better. Detailed results are presented in the Supplementary file.

VI. CONCLUSIONS

This paper proposes FI and AK for evolutionary MOFS for high-dimensional data classification. Based on experiments on 20 real-world datasets, our proposed HMOFS-FI-AK framework can obtain Pareto feature subsets for different scenarios in a time-efficient way. The two techniques are time efficient, simple yet effective, and easy to implement and extend. Experiments on an LLM **TabPFN** for tabular data



(a) (b)

Fig. 8. Friedman test summary of: (a). parameter studies on T in FI and (b). the number of generations for all KNNs used in AK.

verify that HMOFS-FI-AK can obtain effective feature subsets, enabling it to be seamlessly applied to high-dimensional data and achieve superior accuracy and time efficiency.

In the future, it is promising to apply our methods to high-dimensional data in different areas [32] to facilitate the use of LLMs and other advanced techniques.

Please find the Supplementary file for review process in the anonymous GitHub link *https://github.com/Anonymous-cloud-paper/CEC2026-Supplementary-file*.

REFERENCES

[1] H. Saadatmand and M.-R. Akbarzadeh-T, "Many-objective jaccard-based evolutionary feature selection for high-dimensional imbalanced data classification," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 46, no. 12, pp. 8820–8835, 2024.

[2] E. Cheng, D. Doimo, C. Kervadec, I. Macocco, J. Yu, A. Laio, and M. Baroni, "Emergence of a high-dimensional abstraction phase in language transformers," in *International Conference on Learning Representations (ICLR)*, Singapore, Apr 2025.

[3] H. Zhang, Q. Chen, B. Xue, W. Banzhaf, and M. Zhang, "A general feature-informed crossover for two-stage feature selection in symbolic regression," in *2025 IEEE Congress on Evolutionary Computation (CEC)*, 2025, pp. 1–8, doi:10.1109/CEC65147.2025.11043089.

[4] C. Rudin, "Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead," *Nature machine intelligence*, vol. 1, no. 5, pp. 206–215, 2019.

[5] B. Xue, M. Zhang, W. N. Browne, and X. Yao, "A survey on evolutionary computation approaches to feature selection," *IEEE Transactions on Evolutionary Computation*, vol. 20, no. 4, pp. 606–626, 2016.

[6] R. Jiao, B. H. Nguyen, B. Xue, and M. Zhang, "A survey on evolutionary multiobjective feature selection in classification: Approaches, applications, and challenges," *IEEE Transactions on Evolutionary Computation*, vol. 28, no. 4, pp. 1156–1176, 2024.

[7] L. Van Der Maaten, E. O. Postma, H. J. van den Herik *et al.*, "Dimensionality reduction: A comparative review," *Journal of Machine Learning Research*, vol. 10, no. 66-71, p. 13, 2009.

[8] A. Mukhopadhyay, U. Maulik, S. Bandyopadhyay, and C. A. C. Coello, "A survey of multiobjective evolutionary algorithms for data mining: Part i," *IEEE Transactions on Evolutionary Computation*, vol. 18, no. 1, pp. 4–19, 2014.

[9] T. Lu, C. Bian, and C. Qian, "Towards running time analysis of interactive multi-objective evolutionary algorithms," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 38, no. 18, pp. 20 777–20 785, Mar. 2024.

[10] T. Cover and P. Hart, "Nearest neighbor pattern classification," *IEEE Transactions on Information Theory*, vol. 13, no. 1, pp. 21–27, 1967.

[11] S. Han, K. Zhu, M. Zhou, H. Alhumade, and A. Abusorrah, "Locating multiple equivalent feature subsets in feature selection for imbalanced classification," *IEEE Transactions on Knowledge and Data Engineering*, vol. 35, no. 9, pp. 9195–9209, 2023.

[12] H. Xu, B. Xue, and M. Zhang, "A duplication analysis-based evolutionary algorithm for biobjective feature selection," *IEEE Transactions on Evolutionary Computation*, vol. 25, no. 2, pp. 205–218, 2021.

[13] R. Jiao, B. Xue, and M. Zhang, "Learning to preselection: A filter-based performance predictor for multiobjective feature selection in classification," *IEEE Transactions on Evolutionary Computation*, pp. 1–1, 2024.

[14] X. Cai and Y. Xue, "A population initialization method based on similarity and mutual information in evolutionary algorithm for bi-objective feature selection," *ACM Trans. Evol. Learn. Optim.*, vol. 4, no. 3, Jul. 2024.

[15] P. Zhao and L. Lai, "Minimax rate optimal adaptive nearest neighbor classification and regression," *IEEE Transactions on Information Theory*, vol. 67, no. 5, pp. 3155–3182, 2021.

[16] J. Xie, X. Xiang, S. Xia, L. Jiang, G. Wang, and X. Gao, "Mgnr: A multi-granularity neighbor relationship and its application in knn classification and clustering methods," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 46, no. 12, pp. 7956–7972, 2024.

[17] C. E. Shannon, "A mathematical theory of communication," *The Bell system technical journal*, vol. 27, no. 3, pp. 379–423, 1948.

[18] Z. Wang, S. Gao, M. Zhou, S. Sato, J. Cheng, and J. Wang, "Information-theory-based nondominated sorting ant colony optimization for multiobjective feature selection in classification," *IEEE Transactions on Cybernetics*, vol. 53, no. 8, pp. 5276–5289, 2023.

[19] E. Hancer, B. Xue, and M. Zhang, "A multimodal multiobjective evolutionary algorithm for filter feature selection in multilabel classification," *IEEE Transactions on Artificial Intelligence*, vol. 5, no. 9, pp. 4428–4442, 2024.

[20] P. Wang, B. Xue, J. Liang, and M. Zhang, "Multiobjective differential evolution for feature selection in classification," *IEEE Transactions on Cybernetics*, vol. 53, no. 7, pp. 4579–4593, 2023.

[21] Z. Song, H. Wang, B. Xue, and M. Zhang, "Balancing different optimization difficulty between objectives in multiobjective feature selection," *IEEE Transactions on Evolutionary Computation*, vol. 28, no. 6, pp. 1824–1837, 2024.

[22] J. Liang, Y. Zhang, B. Qu, K. Chen, K. Yu, and C. Yue, "A multiform optimization framework for multiobjective feature selection in classification," *IEEE Transactions on Evolutionary Computation*, vol. 28, no. 4, pp. 1024–1038, 2024.

[23] F. Cheng, R. Zhang, Z. Huang, J. Qiu, M. Xia, and L. Zhang, "An objective space constraint-based evolutionary method for high-dimensional feature selection [research frontier]," *IEEE Computational Intelligence Magazine*, vol. 19, no. 2, pp. 113–128, 2024.

[24] N. Hollmann, S. Müller, L. Purucker, A. Krishnakumar, M. Körfer, S. B. Hoo, R. T. Schirrmeister, and F. Hutter, "Accurate predictions on small data with a tabular foundation model," *Nature*, vol. 637, no. 8045, pp. 319–326, 2025.

[25] J. R. Norris, *Markov chains*. Cambridge university press, 1998, no. 2.

[26] H. Chernoff, "A measure of asymptotic efficiency for tests of a hypothesis based on the sum of observations," *The Annals of Mathematical Statistics*, pp. 493–507, 1952.

[27] W. Hoeffding, "Probability inequalities for sums of bounded random variables," *Journal of the American statistical association*, vol. 58, no. 301, pp. 13–30, 1963.

[28] J. He and X. Yao, "Drift analysis and average time complexity of evolutionary algorithms," *Artificial Intelligence*, vol. 127, no. 1, pp. 57–85, 2001.

[29] H. Shastri and E. Frachtenberg, "Revisiting locality in binary-integer representations," *arXiv preprint arXiv:2007.12159*, 2020.

[30] Y. Tian, R. Cheng, X. Zhang, and Y. Jin, "Platemo: A matlab platform for evolutionary multi-objective optimization [educational forum]," *IEEE Computational Intelligence Magazine*, vol. 12, no. 4, pp. 73–87, 2017.

[31] E. Zitzler and L. Thiele, "Multiobjective optimization using evolutionary algorithms—a comparative case study," in *International conference on parallel problem solving from nature*. Springer, 1998, pp. 292–301.

[32] H. T. Ong, E. Karatas, T. Poquillon, G. Grenci, A. Furlan, F. Dilasser, S. B. Mohamad Raffi, D. Blanc, E. Drimaracci, D. Mikec *et al.*, "Digitalized organoids: integrated pipeline for high-speed 3d analysis of organoid structures using multilevel segmentation and cellular topology," *Nature Methods*, pp. 1–12, 2025.