

LLM-FS-Agent: a deliberative role-based large language model architecture for transparent feature selection

Mohamed Bal-Ghaoui¹, Fayssal Sabri²

¹Lab'Innov, Department of Research and Development, Audensiel Conseil, Paris, France

²École Centrale de Lyon, Lyon, France

Article Info

Article history:

Received Oct 21, 2025

Revised Jul 27, 2026

Accepted Aug 12, 2026

Keywords:

Dimensionality reduction

Feature selection

Intrusion detection system

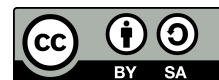
Large language models

Machine learning

ABSTRACT

High-dimensional data can reduce interpretability and increase computational cost in machine learning (ML) pipelines. Large language models (LLMs) offer a promising approach for dimensionality reduction through feature selection (FS), but existing LLM-based methods lack structured reasoning and do not provide clear justifications for their choices. This study propose LLM-FS-Agent, a multi-agent architecture where specialized LLM roles collaborate through a deliberative process to evaluate and rank features. Each selection decision is accompanied by an explicit rationale. We tested our method on the CIC-DIAD 2024 internet of things (IoT) intrusion detection dataset and compared it against LLM-select and traditional techniques like principal component analysis (PCA) across multiple subset sizes. LLM-FS-Agent achieves comparable or better accuracy while reducing classifier training time by 46% on average (statistically significant, 0.094 s, $p = 0.028$ for extreme gradient boosting (XGBoost)). These results show that multi-agent deliberation improves transparency and computational efficiency in FS.

This is an open access article under the [CC BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.



Corresponding Author:

Mohamed Bal-Ghaoui

Lab'Innov, Department of Research and Development, Audensiel Conseil

Paris, France

Email: m.bal-ghaoui@audensiel.com

1. INTRODUCTION

Machine learning (ML) pipelines are central to many scientific and industrial applications. However, their performance is often limited by curse of dimensionality [1]. As the number of features grows, computational costs increase, overfitting becomes more likely, and generalization degrades [2]. Addressing this issue is important for reliable ML deployment. Feature selection (FS) aims to find a small set of informative features while keeping their original meaning. Traditional FS methods are grouped into filter, wrapper, and embedded approaches [3]–[5]. Filter methods rank features using statistical criteria without involving a learning model [6]. Wrapper methods search for feature subsets that maximize predictive performance [7]. Embedded methods integrate FS into model training through regularization [8], [9]. These methods work well but often behave as black boxes. They provide little explanation for why specific features are selected or discarded. This lack of transparency is especially problematic in cybersecurity, where analysts need to understand and trust the decisions made by ML systems.

Large language models (LLMs) have created new opportunities for FS. Li *et al.* [10] classify LLM-based FS into text-based and data-driven approaches. Text-based FS uses only feature names and task descriptions, relying on the LLMs pre-trained knowledge without needing numerical data. This

works well when data is scarce and improves with larger models. The LLM-select framework [11] proposes several prompting strategies (LLM-score, LLM-rank, LLM-SEQ) and shows that LLMs can match classical FS methods even without training data. However, performance depends heavily on model size. Retrieval-augmented feature selection (RAFS) [10] adds external domain knowledge to improve results in specialized fields like medicine. Data-driven FS gives LLMs actual feature values or sample data so they can perform statistical reasoning [10]. This works in limited settings but is constrained by LLM context windows. As sample size grows, performance degrades [12], [13]. Data-driven methods often underperform text-based approaches in both low-resource and full-shot settings. LLMs have also been used for feature engineering. Han *et al.* [14] used LLMs to generate meta-features, and Yang *et al.* [15] proposed in-context evolutionary search (ICE-SEARCH), which refines feature subsets based on test performance.

The limits of single-agent prompting have led to multi-agent LLM architectures that handle complex tasks through modular reasoning and coordination [16], [17]. These systems connect LLMs to external tools and application programming interfaces (APIs) for structured problem solving [18], [19]. Applications in data science show promise for automated analysis [20], [21]. Frameworks like tree-of-thought (ToT) [22], graph-of-thought (GoT) [23], and mixture-of-agents (MoA) [24] show that coordinated multi-agent reasoning can outperform single models. These methods improve reliability and interpretability through structured deliberation. However, existing multi-agent frameworks are designed for general reasoning and lack the specific mechanisms needed for FS. They do not address issues like feature redundancy, collinearity, or robustness against adversarial manipulation, which are critical in cybersecurity.

2. METHOD

2.1. Experimental setup and data preprocessing

All experiments were conducted using the CIC-DIAD 2024 dataset [25], which was developed for internet of things (IoT) intrusion detection and comprises 84 features. This study considered three classes namely benign, brute force, and mirai. The preprocessing pipeline involved data cleaning, the removal of highly correlated features [3] based on a Pearson correlation threshold greater than 0.9, and feature standardization using StandardScaler. To address the class imbalance issue, random undersampling was applied, resulting in a balanced dataset. Table 1 presents the class distribution before and after the preprocessing steps.

Table 1. Dataset distribution before and after preprocessing

Class/Dataset	Original	Preprocessed
Benign	183 595 (94.0%)	3 619 (29.2%)
Mirai	5 170 (2.6%)	5 170 (41.7%)
Brute Force	3 619 (1.9%)	3 619 (29.2%)
Total	192,384 (100%)	12,408 (100%)

2.2. Evaluation protocol

Figure 1 shows the evaluation workflow adopted in this study. Four classical ML classifiers were selected namely random forest (RF), extreme gradient boosting (XGBoost), support vector classifier (SVC), and logistic regression (LR). These models represent different learning paradigms, including ensemble learning, gradient boosting, kernel-based classification, and linear modeling [2]. Model performance was evaluated using accuracy and the area under the curve (AUC) receiver operating characteristic (ROC). All LLMs were executed locally using the Ollama framework, including Llama 3.2 (2 B), Gemma_3 (4B), Qwen (4B), Phi-3 Mini (3B), and Mistral (4B). Based on the generated feature importance rankings, feature subsets of different sizes were evaluated, with $n \in 5, 10, 20, 30, 40$, and 50. These experiments were designed to investigate the impact of feature subset size, FS method, LLM configuration, and classifier choice on the overall performance.

Additional experiments were conducted using smaller models, including Gemma3-1B, Qwen3-0.6B, LLaMA-3.2-1B, and Phi-3-Mini, to further analyze the influence of model capacity and role assignment within the proposed framework. For each experiment, a ranked feature list was generated. The refiner role uses statistical guidance by analyzing feature-level metadata, including feature mean, standard deviation, and correlation with the target variable. This information provides additional evidence for refining the initial feature importance scores produced by the initiator role, enabling a more informed feature ranking process.

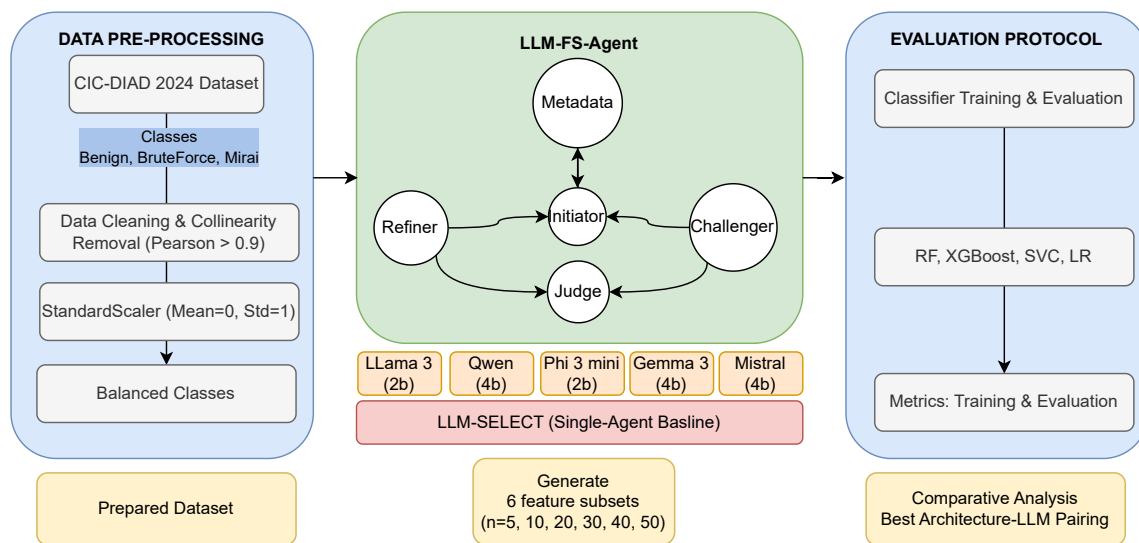


Figure 1. Pipeline and evaluation protocol of LLM-FS-Agent

2.3. LLM-FS-Agent architecture design and the deliberative feature selection process

The proposed LLM-FS-Agent is a transparent multi-agent FS framework [16], [17] that employs a structured deliberation mechanism [22]–[24]. As illustrated in Figure 2, the framework receives feature names and a task description as inputs [10]. Each role-specific agent generates a set of arguments that either justify or question the relevance of individual features, as summarized in Table 2. To facilitate reproducibility, the prompts associated with each agent role are provided in Figures 3 to 6. These prompts specify the instructions, contextual information, and expected outputs for each stage of the deliberative FS process.

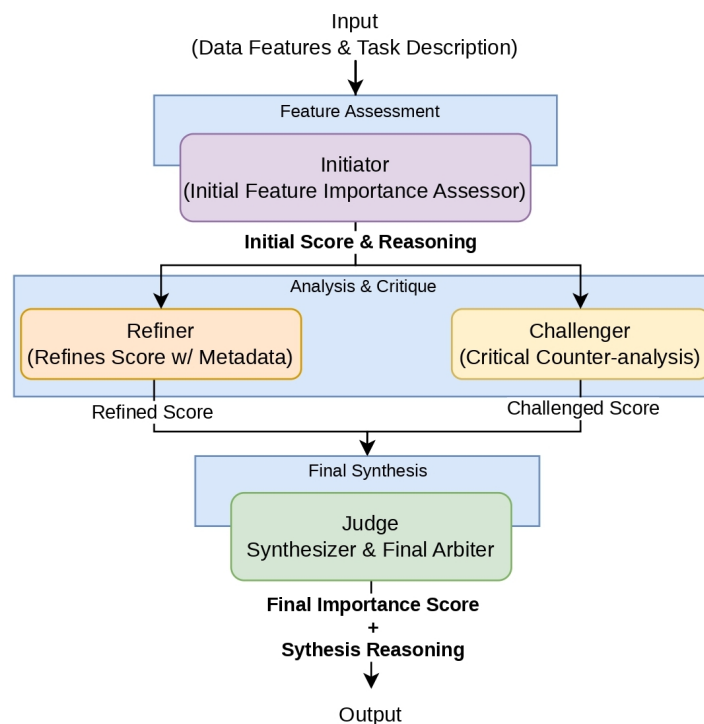


Figure 2. Architecture of the LLM-FS-Agent

Table 2. Roles and responsibilities of agents in the LLM-FS-Agent framework

Agent role	Description
Initiator	Does an initial semantic analysis of each feature based on the task description and provides a first relevance score.
Refiner	Improves the Initiator's analysis by adding supporting arguments and statistical metadata, such as mean, standard deviation, and feature-target correlation.
Challenger	Reviews the Initiator's arguments critically to find weaknesses, redundancies, or biases, and provides counter-arguments in a peer-review style.
Judge	Acts as the final decision maker by combining all arguments. It computes a final importance score (S_{final}) for each feature as a weighted sum of the refined (S_{refined}) and challenged ($S_{\text{challenged}}$) scores:

$$S_{\text{final}} = w_r S_{\text{refined}} + w_c S_{\text{challenged}}, \quad w_r + w_c = 1$$

Prompt 1 – Initiator: Semantic-Only Initial Evaluation

Your role is to perform an initial assessment of the importance of a feature for a prediction task.

Task: task_description

Feature to evaluate: feature_name

Provide an initial importance score between 0.0 and 1.0 and a concise reasoning. THE REASONING MUST BE A SINGLE SENTENCE. Your answer MUST be a single JSON code block.

```
{
  "score": 0.0,
  "reasoning": "Initial reasoning based on the semantics of the feature."
}
```

Figure 3. The initiator LLM prompt template

Prompt 2 – Refiner: Metadata-Aware Refinement

Your role is to refine an initial analysis by integrating statistical metadata.

Task: {task_description}

Feature: {feature_name}

Initial Score: {initial_score}

Initial Reasoning: "{initial_reasoning}"

Statistical metadata for this feature: {metadata}

Refine the score and reasoning based on this new information. Your answer MUST be a single JSON code block.

```
{
  "refined_score": 0.0,
  "refined_reasoning": "Refined reasoning incorporating the metadata."
}
```

Figure 4. The refiner LLM prompt template

Prompt 3 – Challenger: Critical Review

Your role is to take a critical viewpoint on a feature importance analysis.

Task: {task_description}

Feature: {feature_name}

Initial Score: {initial_score} Initial Reasoning: "{initial_reasoning}"

Question this reasoning with ONE OR TWO KEY COUNTERARGUMENTS.

Propose your own score (between 0.0 and 1.0), which may confirm or dispute the initial score. Your answer MUST be a single JSON code block.

```
{
  "challenged_score": 0.0,
  "challenged_reasoning": "Critical reasoning and counter-arguments."
}
```

Figure 5. The challenger LLM prompt template

Prompt 4 – Judge: Final Arbitration

Your role is to act as an expert arbiter to synthesize two divergent analyses and produce a final evaluation.

ANALYSIS A (from the Quantitative Refiner)
Proposed Score: {score_Refiner}
Reasoning: "{reasoning_Refiner}"

ANALYSIS B (from the Critical Challenger)
Proposed Score: {score_Challenger}
Reasoning: "{reasoning_Challenger}"

YOUR INSTRUCTIONS:

- Compare the reasonings for their strength and relevance.
- Determine weights w_A (for Analysis A) and w_B (for Analysis B), both between 0 and 1, that reflect your confidence in each analysis, subject to the constraint:
$$w_A + w_B = 1$$
- Justify your weights by explaining why you give more credit to one analysis over the other.
- Calculate the final score using the formula:
$$\text{Final_Score} = (w_A \times \{\text{score_Refiner}\}) + (w_B \times \{\text{score_Challenger}\})$$
- Write a synthesis reasoning that explains the final verdict (IN THREE SENTENCES MAXIMUM).

Your answer MUST be a single JSON code block.

```
{
  "final_score": 0.65,
  "weights": {
    "w_A": 0.4,
    "w_B": 0.6
  },
  "weights_justification": "I assigned these weights because...",
  "final_reasoning": "Final reasoning of the arbiter."
}
```

Figure 6. The judge LLM prompt template

3. RESULTS AND DISCUSSION

3.1. Quantitative performance and generalizability

The performance of LLM-FS-Agent was compared with LLM-select [11] and principal component analysis (PCA) [3] using downstream classifiers, including XGBoost (Figure 7), RF (Figure 8), SVC (Figure 9), and LR (Figure 10), across different feature subset sizes ($n = 5, 10, 20, 30, 40$, and 50). As illustrated in Figures 7 and 8, XGBoost and RF consistently achieve high accuracy across varying subset sizes, demonstrating the robustness of the selected feature representations. Table 3 summarizes the mean AUC and accuracy values obtained at the best-performing subset size ($n = 20$). Under this configuration, LLM-FS-Agent reduces performance variability compared with single-agent FS approaches and improves accuracy by up to +1.49% in cases where LLM-select performs less effectively, while introducing only marginal decreases for the remaining models. Figure 11 provides a comprehensive comparison across the six evaluated feature subset sizes, where Figures 11(a) to 11(f) report accuracy results and Figures 11(g) to 11(l) present AUC measurements. Regarding accuracy, LLM-select achieves competitive results under the most constrained feature size ($n = 5$), whereas LLM-FS-Agent shows increasing advantages as the subset size grows, reaching its best performance at $n = 20$. Although LLM-select remains comparable at $n = 40$, it exhibits larger performance fluctuations, whereas LLM-FS-Agent maintains stable results across larger feature subsets. A similar trend is observed for AUC, while all approaches provide strong discriminative capability with only five selected features, LLM-FS-Agent delivers more consistent performance across $n = 10$ – 50 , with $n = 20$ providing the most stable results across different classifiers and LLM backbones. Overall, LLM-FS-Agent achieves superior performance for intermediate and larger subsets ($n = 10, 20, 30$, and 50), while LLM-Select remains competitive mainly at $n = 5$ and occasionally at $n = 40$. These findings suggest that the multi-agent framework enables more reliable feature ranking across different feature budgets.

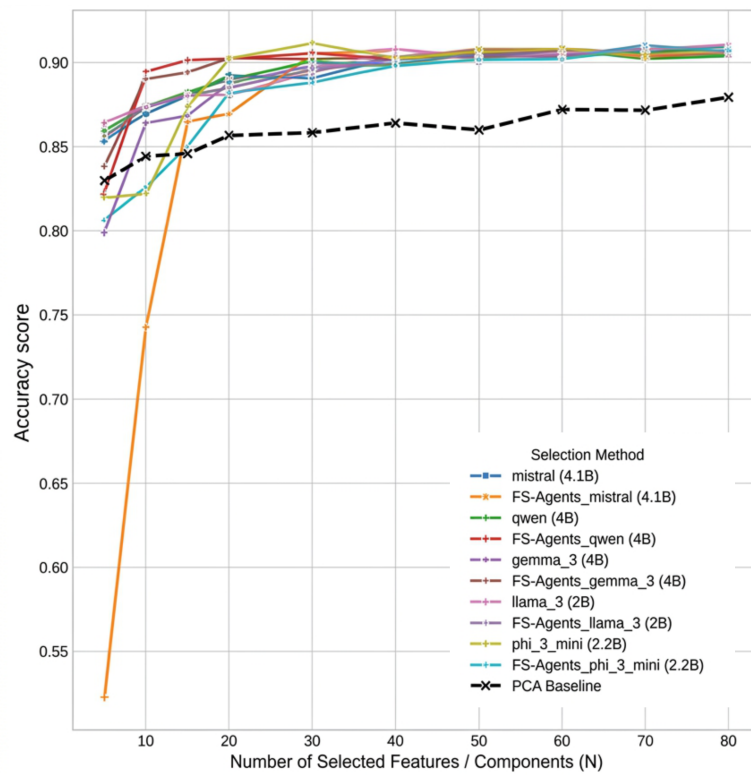


Figure 7. Performance curve of the XGBoost classifier

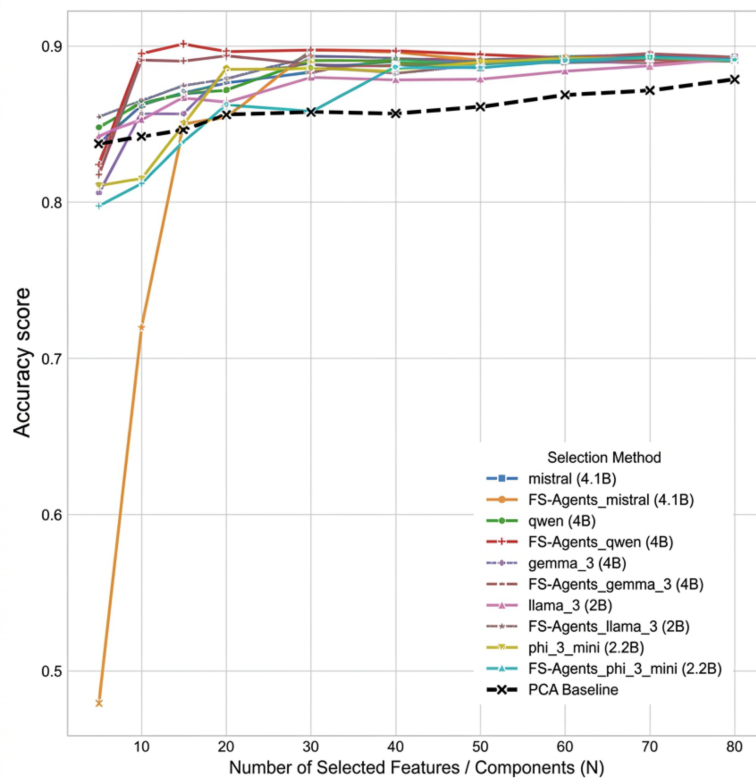


Figure 8. Performance curve of the RF classifier

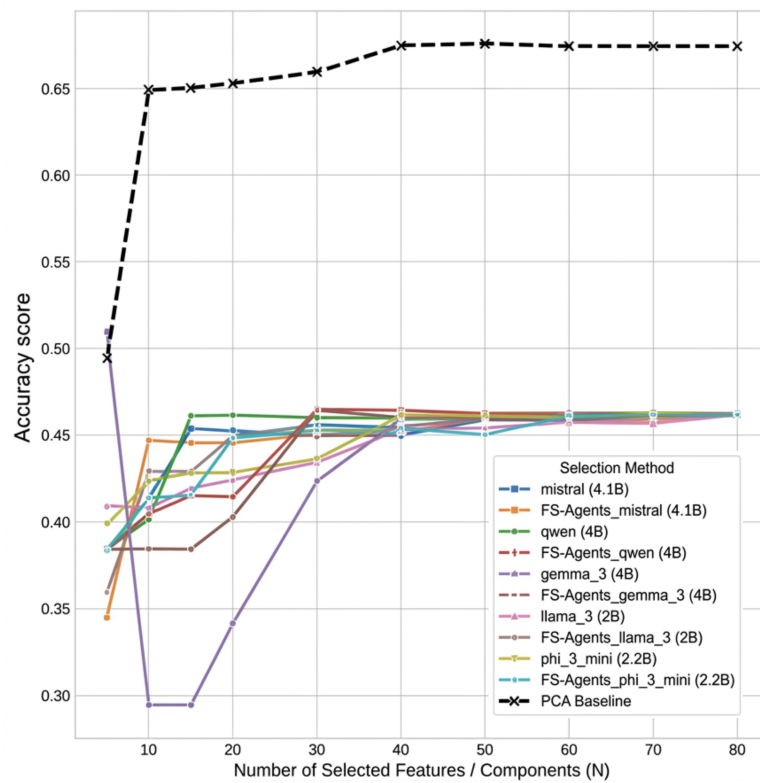


Figure 9. Performance curve of the SVC

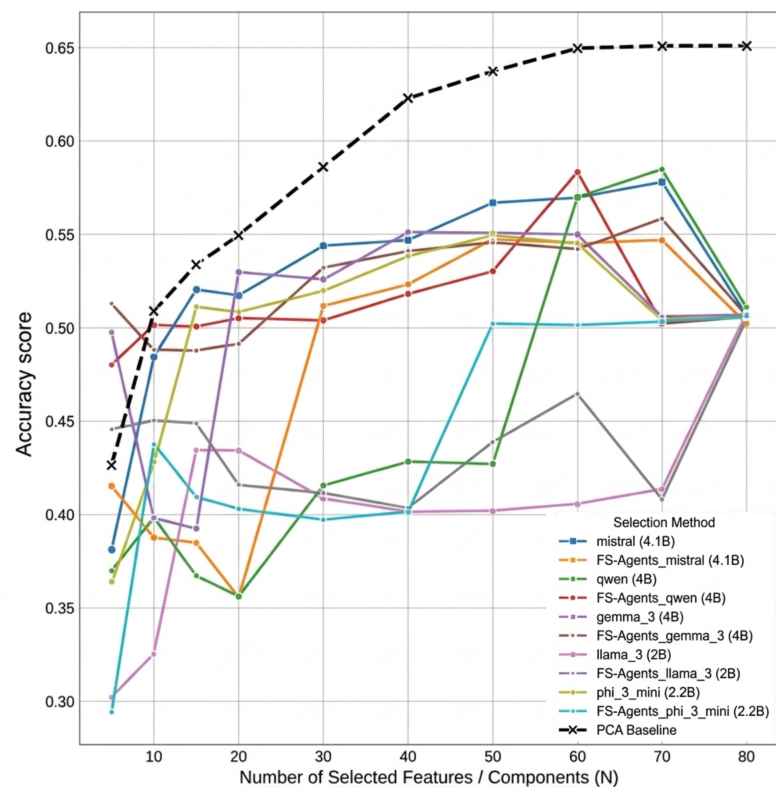


Figure 10. Performance curve of the LR classifier

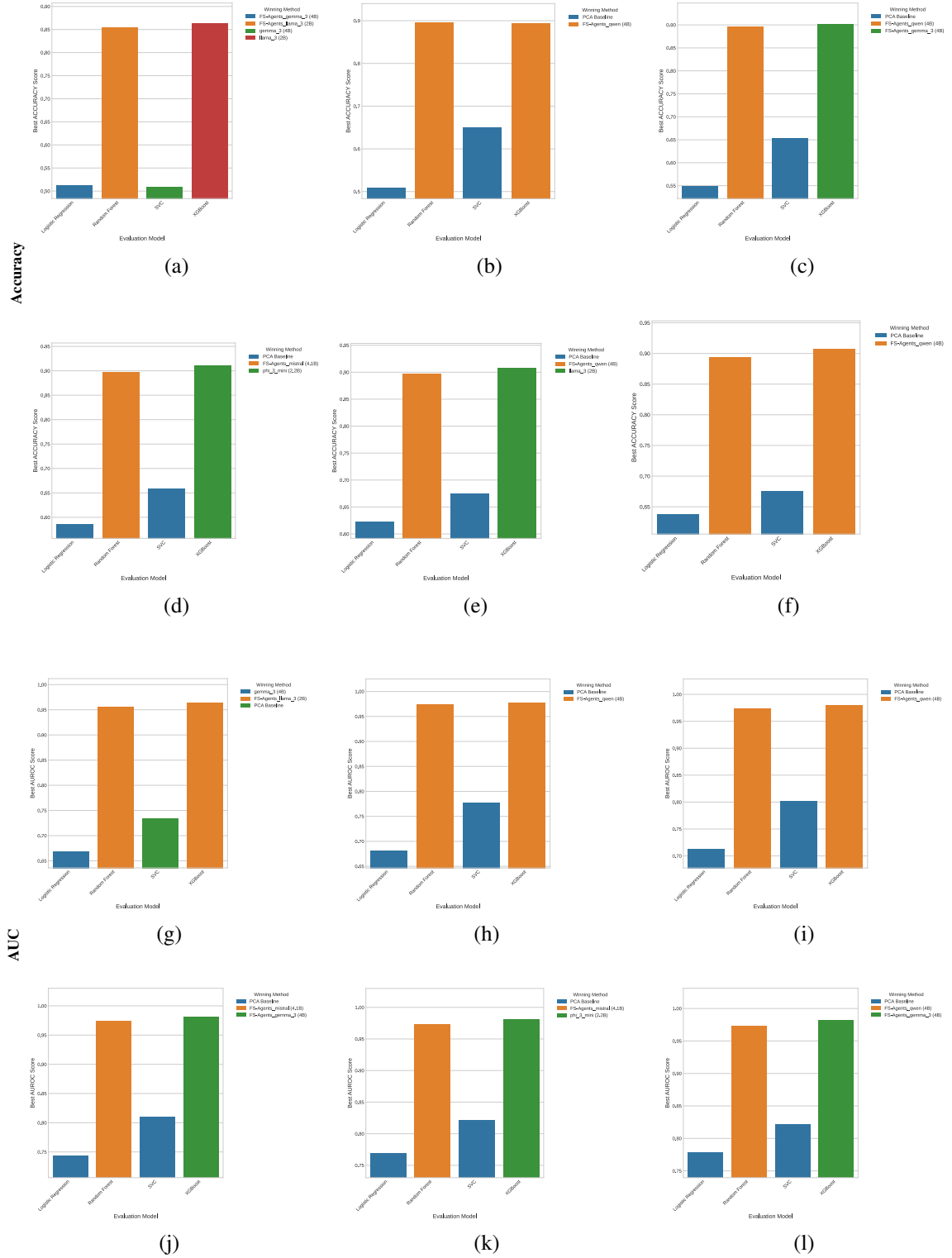


Figure 11. Comparison of accuracy and AUC across feature subset sizes: (a) accuracy at $n = 5$, (b) accuracy at $n = 10$, (c) accuracy at $n = 20$, (d) accuracy at $n = 30$, (e) accuracy at $n = 40$, (f) accuracy at $n = 50$; (g) AUC at $n = 5$, (h) AUC at $n = 10$, (i) AUC at $n = 20$, (j) AUC at $n = 30$, (k) AUC at $n = 40$, and (l) AUC at $n = 50$

Table 3. Performance comparison of best models on LLM-Select vs. LLM-FS-Agent (average across subsets)

LLM	AUC				$\Delta\%$	Accuracy				$\Delta\%$
	LLM-select		LLM-FS-Agent			LLM-select		LLM-FS-Agent		
	XGB	RF	XGB	RF		XGB	RF	XGB	RF	
Mistral	0.976	0.964	0.965	0.947	-0.99	0.892	0.876	0.870	0.854	-2.25
Qwen	0.975	0.963	0.980	0.974	+0.46	0.890	0.872	0.902	0.897	+1.21
Gemma3-4B	0.973	0.968	0.978	0.973	+0.50	0.888	0.884	0.903	0.894	+1.49
LLaMA3-2B	0.973	0.962	0.974	0.964	+0.15	0.880	0.864	0.885	0.879	+0.45
Phi3-mini	0.979	0.969	0.972	0.962	-0.64	0.902	0.886	0.882	0.862	-1.97
Average	0.975	0.965	0.974	0.964	+0.10	0.890	0.876	0.888	0.877	+0.19

3.2. Computational efficiency and qualitative evaluation of feature reasoning

Analysis of Tables 4 and 5 shows that LLM-FS-Agent reduces downstream classifier training time. For XGBoost, the average speedup is $1.87\times$ with statistical significance ($p = 0.028$, Cohen's $d = 0.87$). Inference times improved slightly. The gains in AUC and accuracy, while modest, support the multi-agent framework and offset the deliberation overhead.

Table 4. Computational efficiency comparison (XGBoost)

LLM backbone	Training time (s)		Inference time (s)		Speedup	
	LLM-select	LLM-FS-Agent	LLM-select	LLM-FS-Agent	Training	Inference
Mistral	0.434	0.108	0.0023	0.0016	$4.02\times$	$1.44\times$
Qwen	0.117	0.108	0.0014	0.0015	$1.08\times$	$0.93\times$
Gemma3-4B	0.101	0.113	0.0015	0.0016	$0.89\times$	$0.94\times$
LLaMA3-2B	0.252	0.108	0.0018	0.0015	$2.33\times$	$1.20\times$
Phi3-mini	0.121	0.117	0.0016	0.0017	$1.03\times$	$0.94\times$
Average	0.205	0.111	0.0017	0.0016	$1.87\times$	$1.09\times$

Table 5. Statistical significance analysis (XGBoost)

Metric	Mean Diff.	p-value	Cohen's d	Effect
AUC	+0.0010	0.342	0.28	Small
Accuracy	+0.0017	0.456	0.19	Small
Training time	-0.094s	0.028*	-0.87	Large
Inference time	-0.0001s	0.234	-0.31	Small

*Statistically significant at $\alpha = 0.05$

Effect size: $|d| < 0.2$ (negligible), 0.2-0.5 (small), 0.5-0.8 (medium), > 0.8 (large)

3.3. Statistical guidance and large language model role effects on feature selection

The impact of statistical guidance and LLM role configurations on FS was evaluated through downstream ML performance, including confusion matrices and overall accuracy. Using LLM-FS-Agent with four LLM backbones, all 71 features were analyzed and ranked according to their estimated importance. Table 6 presents the feature importance scores generated by each LLM role, along with the final aggregated scores produced by the judge agent across the four best-performing experiments. Temporal and packet-level flow features, such as packet length min/max, Fwd inter-arrival time (IAT) mean/max/Std, and idle min, consistently appear among the highest-ranked features. Their frequent selection indicates strong discriminative capability for distinguishing benign and malicious traffic patterns. In addition, source and destination port statistics, as well as flag-count features, are repeatedly selected, reflecting their relevance from a network traffic analysis perspective [25]. The consistency among different LLM roles and the stability of the top-ranked features suggest that these indicators provide reliable information for intrusion detection. Table 7 summarizes the four best-performing ML configurations. XGBoost achieved the strongest overall performance, obtaining accuracy values ranging from 0.876 to 0.886, F1-scores between 0.873 and 0.884, and AUC values from 0.971 to 0.975, while maintaining moderate training requirements and efficient inference. RF achieved slightly lower performance, with accuracy ranging from 0.866 to 0.881, F1-scores from 0.863 to 0.879, and AUC values between 0.962 and 0.971; however, it provided faster training, making it a practical alternative in resource-constrained scenarios. In contrast, SVC and LR exhibited lower predictive performance, with SVC additionally incurring higher training and inference costs. Among the LLM role configurations, experiment 1 achieved the best downstream classification results, indicating that the assignment and interaction of different agent roles can significantly influence the quality of generated feature rankings and, consequently, the final model performance.

Table 6. Top-10 features ranked by framework scores

Experiment	Feature	Initiator	Refiner	Challenger	Judge	Judge Reasoning
Experiment 1	Packet length min	0.850	0.800	0.200	0.900	Analyzing packet length patterns helps distinguish benign from malicious traffic; outliers and standard deviation considered.
	Fwd IAT mean	0.850	1.000	0.800	0.900	Statistically significant patterns in forward IAT are key for threat detection; false positives must be considered.
	Src port	0.900	0.800	0.800	0.800	Source port may indicate anomalies when correlated with other indicators; useful for initial threat assessment.
	Fwd IAT total	0.850	1.000	0.200	0.800	Forward IAT Total can indicate anomalous traffic; should not be used in isolation for prediction.
	Subflow bwd packets	0.850	0.800	0.200	0.800	Subflow backward packets patterns can indicate malicious propagation; statistical significance prioritized.
	Bwd IAT mean	0.900	0.800	0.200	0.760	Useful for detecting network anomalies; dynamic threat environment requires cautious interpretation.
	Bwd packets/s	0.850	0.850	0.200	0.750	Provides network performance insight; must be contextualized to distinguish benign from malicious patterns.
	FWD init win bytes	0.850	0.850	0.200	0.740	Forward initial window bytes can predict malicious activity; statistical evidence prioritized.
	Active max	0.850	0.900	0.300	0.720	High activity levels indicate potential threats; should be used with other features for accuracy.
	Flow IAT std	0.850	0.850	0.200	0.720	Standard deviation of flow IAT can signal malicious behavior; context with other indicators is critical.
Experiment 2	Active mean	0.65	0.85	0.50	0.85	Domain-focused reasoning: correlation with malicious activity and traffic patterns.
	RST flag count	0.65	0.80	0.50	0.80	Provides clear indication of potential malicious activity, critical for domain-focused predictions.
	Fwd IAT std	0.65	0.80	0.50	0.80	Strong statistical significance; considered for predicting malicious behavior.
	Src port	0.65	0.85	0.50	0.75	Contextual information such as mean and std supports reliability as an indicator.
	Flow bytes/s	0.85	0.95	0.50	0.75	Key indicator of network activity; essential for tasks involving data movement.
	Packet length mean	0.65	0.82	0.50	0.75	Important for detecting malicious activity, but context-dependent.
	Idle min	0.65	0.85	0.70	0.75	Quantitative strength in predicting malicious activity; secondary importance noted.
	Protocol	0.85	0.92	0.60	0.72	Relevant for prediction tasks; other features may provide redundancy.
	Flow IAT max	0.65	0.80	0.70	0.70	Strong correlation with malicious activity; critical for predictive tasks.
	Subflow fwd packets	0.65	0.85	0.50	0.70	Prioritized as indicative of malicious traffic in network analysis.
Experiment 3	Packet length min	0.850	0.700	0.850	0.700	Helps distinguish normal vs malicious traffic; outliers and standard deviation important.
	Flow IAT max	0.850	0.790	0.700	0.700	Important, but quantitative analysis suggests limitations due to normal traffic patterns.
	Idle min	0.850	0.670	0.600	0.675	High idle times indicate potential malicious activity; domain reasoning prioritizes this feature.
	Packet length variance	0.850	0.720	0.500	0.660	Variability in packet lengths may indicate anomalies; less predictive alone.
	Dst Port	0.850	0.720	0.500	0.650	Captures diverse usage; not primary indicator but useful in combination with other features.
	Active Std	0.900	0.750	0.500	0.625	Indicates variability often accompanying malicious activity; importance context-dependent.
	Subflow bwd packets	0.850	0.670	0.500	0.625	Rapid propagation patterns indicative of malicious traffic; more relevant in large networks.
	Total bwd packets	0.850	0.670	0.600	0.625	Low variance reduces standalone importance; domain-focused reasoning supports use.
	Packet length max	0.850	0.620	0.700	0.625	Standard deviation and context considered; moderately informative predictor.
	FIN flag count	0.850	0.620	0.750	0.620	Quantitative analysis emphasizes statistical significance; useful but less critical.
Experiment 4	Packet length max	0.800	0.760	0.600	0.720	Important for identifying malicious traffic considering thresholds and context.
	Fwd seg size min	0.750	0.730	0.600	0.690	Relatively high mean size dispersion; indicative of malicious traffic in combination with other features.
	FWD init win bytes	0.850	0.670	0.700	0.670	Weak correlation with target; contextual importance slightly more relevant than raw stats.
	Bwd IAT min	0.850	0.760	0.600	0.667	Low correlation with network activity; unlikely to be decisive alone.
	Fwd IAT mean	0.850	0.670	0.600	0.667	Distribution patterns considered; correlation with other features moderately informative.
	Packet length min	0.800	0.600	0.600	0.667	Useful for anomaly detection; simplicity reduces standalone predictive power.
	Bwd packet length max	0.800	0.650	0.600	0.650	Captures suspicious activity patterns; statistical metadata moderately informative.
	Flow packets/s	0.750	0.670	0.700	0.625	Low correlation alone; contextual factors increase relevance.
	Bwd packet length min	0.850	0.650	0.600	0.625	Strong semantic relevance; moderate correlation with target variable.
	Idle min	0.850	0.750	0.600	0.625	Some predictive power; may be noisy, but domain-focused reasoning supports use.

Table 7. ML performances across top-performing experiments

Experiment (Initiator/refiner/challenger/judge)	Model	Accuracy	Precision	Recall	F1	AUC	Train (s)	Inference (s)
Experiment 1: gemma3:1b / qwen3:0.6b / llama3.2:1b / phi3:mini	XGBoost	0.886	0.885	0.884	0.884	0.975	1.728	0.014
	RF	0.881	0.881	0.879	0.879	0.971	0.351	0.057
	SVC	0.624	0.654	0.604	0.609	0.759	10.397	1.319
	LR	0.510	0.519	0.500	0.496	0.667	0.020	0.001
Experiment 2: phi3:mini / gemma3:1b / qwen3:0.6b / llama3:2.1b	XGBoost	0.879	0.879	0.877	0.877	0.971	1.562	0.013
	RF	0.874	0.874	0.871	0.872	0.967	0.340	0.058
	SVC	0.599	0.615	0.576	0.577	0.737	10.048	1.356
	LR	0.496	0.493	0.487	0.482	0.679	0.024	0.001
Experiment 3: gemma3:1b / phi3:mini / qwen3:0.6b / llama3:1b	XGBoost	0.877	0.875	0.875	0.875	0.972	1.601	0.013
	RF	0.866	0.864	0.861	0.863	0.962	0.323	0.058
	SVC	0.606	0.645	0.574	0.580	0.751	10.567	1.337
	LR	0.565	0.571	0.534	0.534	0.681	0.036	0.001
Experiment 4: qwen3:0.6b / phi3:mini / gemma3:1b / llama3:mini	XGBoost	0.876	0.875	0.873	0.873	0.972	1.578	0.012
	RF	0.868	0.866	0.864	0.865	0.964	0.312	0.061
	SVC	0.584	0.601	0.564	0.570	0.731	10.883	1.423
	LR	0.420	0.443	0.433	0.421	0.620	0.016	0.001

Figure 12 details the classification outcomes for the XGBoost model. The prominent values located along the main diagonal underscore the model's discriminative power, correctly grouping 3,070 benign flows, 3,259 brute force attacks, and 4,667 mirai instances. The off-diagonal errors remain relatively constrained, though 341 benign samples are misclassified as mirai, and 266 mirai samples are misidentified as benign; the remaining pairwise errors are below 240 instances. These specific error patterns, when taken alongside the strong diagonal counts, substantiate the proposed integration of ensemble learning with multi-role LLM-guided FS, producing an intrusion detection system that is both numerically reliable and inherently interpretable.

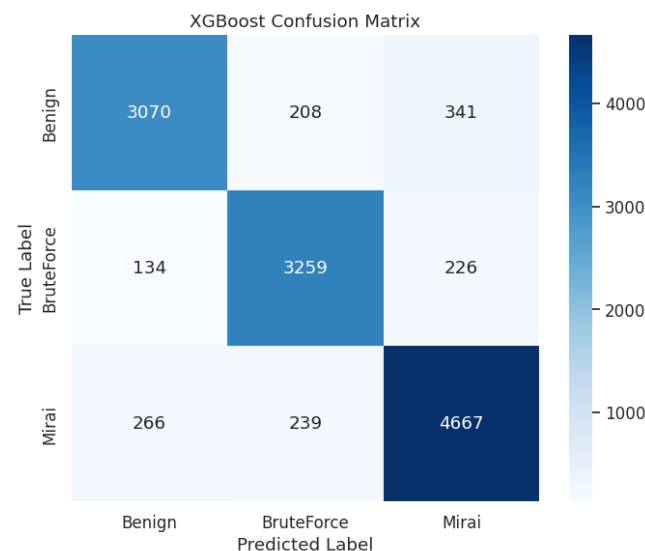


Figure 12. Confusion matrix for the best model

3.4. Limitations and future directions

Despite the effectiveness demonstrated by the proposed multi-agent framework, several limitations remain. Although different lightweight LLMs were assigned to specialized roles to promote diverse reasoning perspectives, the framework may still be affected by the selection of underlying LLMs. Variations in model architecture, capability, or configuration may result in different feature prioritization outcomes [11]. Future research will investigate adaptive mechanisms for dynamically adjusting agent weights (w_r , w_c) according to feature complexity, uncertainty, or contextual characteristics. Furthermore, the evaluation will be extended to a wider range of cybersecurity scenarios to further examine the scalability, robustness, and generalizability of the proposed approach.

4. CONCLUSION

This work introduced LLM-FS-Agent, a multi-agent FS framework designed to improve the interpretability and accountability of FS processes. By coordinating role-specialized LLM agents, the proposed framework generates human-interpretable rationales for feature rankings, enhancing the transparency of FS decisions. Experimental results on an IoT intrusion detection dataset demonstrate that LLM-FS-Agent can identify effective feature subsets, achieving near-optimal performance with $n = 20$ selected features while maintaining competitive results across different classifiers and evaluation metrics. These findings highlight the potential of multi-agent LLM-based FS as an effective, robust, and interpretable approach for cybersecurity applications.

ACKNOWLEDGMENTS

The authors gratefully acknowledge Lab’Innov at Audensiel Department of R and D for providing the research environment and technical support that enabled this work. This study forms part of the advancement of the internal research and development project STAAC.

FUNDING INFORMATION

Authors state no funding involved.

AUTHOR CONTRIBUTIONS STATEMENT

This journal uses the Contributor Roles Taxonomy (CRediT) to recognize individual author contributions, reduce authorship disputes, and facilitate collaboration.

Name of Author	C	M	So	Va	Fo	I	R	D	O	E	Vi	Su	P	Fu
Mohamed Bal-Ghaoui	✓	✓		✓	✓	✓	✓	✓		✓	✓	✓	✓	
Fayssal Sabri			✓			✓		✓	✓					

C : Conceptualization

M : Methodology

So : Software

Va : Validation

Fo : Formal Analysis

I : Investigation

R : Resources

D : Data Curation

O : Writing - Original Draft

E : Writing - Review & Editing

Vi : Visualization

Su : Supervision

P : Project Administration

Fu : Funding Acquisition

CONFLICT OF INTEREST STATEMENT

Authors state no conflict of interest.

DATA AVAILABILITY

Data availability does not apply to this paper as no new data were created or analyzed in this study.

REFERENCES

- [1] M. Verleysen and D. François, “The curse of dimensionality in data mining and time series prediction,” in *Computational Intelligence and Bioinspired Systems*, vol. 3512, 2005, pp. 758–770, doi: 10.1007/11494669_93.
- [2] P. Domingos, “A few useful things to know about machine learning,” *Communications of the ACM*, vol. 55, no. 10, pp. 78–87, Oct. 2012, doi: 10.1145/2347736.2347755.
- [3] R. Zebari, A. Abdulazeez, D. Zeebaree, D. Zebari, and J. Saeed, “A comprehensive review of dimensionality reduction techniques for feature selection and feature extraction,” *Journal of Applied Science and Technology Trends*, vol. 1, no. 1, pp. 56–70, May 2020, doi: 10.38094/jastt1224.
- [4] G. Chandrashekar and F. Sahin, “A survey on feature selection methods,” *Computers & Electrical Engineering*, vol. 40, no. 1, pp. 16–28, Jan. 2014, doi: 10.1016/j.compeleceng.2013.11.024.
- [5] J. Tang, S. Alelyani, and H. Liu, “Feature selection for classification: a review,” in *Data Classification: Algorithms and Applications*. Boca Raton, United States: Chapman and Hall/CRC, 2014, pp. 65–92, doi: 10.1201/b17320-7.
- [6] M. Cherrington, F. Thabtah, J. Lu, and Q. Xu, “Feature selection: filter methods performance challenges,” in *2019 International Conference on Computer and Information Sciences (ICCIS)*, Apr. 2019, pp. 1–4, doi: 10.1109/ICCISci.2019.8716478.

- [7] N. E. Aboudi and L. Benhlima, "Review on wrapper feature selection approaches," in *2016 International Conference on Engineering & MIS (ICEMIS)*, Sep. 2016, pp. 1–5, doi: 10.1109/ICEMIS.2016.7745366.
- [8] R. Tibshirani, "Regression shrinkage and selection via the lasso," *Journal of the Royal Statistical Society Series B: Statistical Methodology*, vol. 58, no. 1, pp. 267–288, Jan. 1996, doi: 10.1111/j.2517-6161.1996.tb02080.x.
- [9] M. Yuan and Y. Lin, "Model selection and estimation in regression with grouped variables," *Journal of the Royal Statistical Society Series B: Statistical Methodology*, vol. 68, no. 1, pp. 49–67, Feb. 2006, doi: 10.1111/j.1467-9868.2005.00532.x.
- [10] D. Li, Z. Tan, and H. Liu, "Exploring large language models for feature selection: a data-centric perspective," *ACM SIGKDD Explorations Newsletter*, vol. 26, no. 2, pp. 44–53, 2025.
- [11] D. P. Jeong, Z. C. Lipton, and P. Ravikumar, "LLM-Select: feature selection with large language models," *Transactions on Machine Learning Research*, pp. 1–77, 2025.
- [12] Z. Dong, T. Tang, J. Li, W. X. Zhao, and J.-R. Wen, "BAMBOO: a comprehensive benchmark for evaluating long text modeling capacities of large language models," in *2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, May 2024, pp. 2086–2099.
- [13] N. F. Liu *et al.*, "Lost in the middle: how language models use long contexts," *Transactions of the Association for Computational Linguistics*, vol. 12, pp. 157–173, Feb. 2024, doi: 10.1162/tacl.a.00638.
- [14] S. Han, J. Yoon, S. O. Arik, and T. Pfister, "Large language models can automatically engineer features for few-shot tabular learning," in *41st International Conference on Machine Learning*, Jul. 2024, pp. 17454–17479.
- [15] T. Yang, T. Yang, F. Lyu, S. Liu, and X. Liu, "ICE-SEARCH: a language model-driven feature selection approach," 2024, *arXiv:2402.18609*.
- [16] L. Wang *et al.*, "A survey on large language model based autonomous agents," *Frontiers of Computer Science*, vol. 18, no. 6, Dec. 2024, doi: 10.1007/s11704-024-40231-1.
- [17] Z. Xi *et al.*, "The rise and potential of large language model based agents: a survey," *Science China Information Sciences*, vol. 68, no. 2, Feb. 2025, doi: 10.1007/s11432-024-4222-0.
- [18] R. Yang *et al.*, "GPT4Tools: teaching large language model to use tools via self-instruction," *37th International Conference on Neural Information Processing Systems*, 2023, pp. 71995–72007, doi: 10.52202/075280-3149.
- [19] T. Schick *et al.*, "Toolformer: language models can teach themselves to use tools," in *37th International Conference on Neural Information Processing Systems*, pp. 68539–68551, 2023, doi: 10.52202/075280-2997.
- [20] S. Hong *et al.*, "Data interpreter: an LLM agent for data science," in *Findings of the Association for Computational Linguistics: ACL 2025*, Jul. 2025, pp. 19796–19821, doi: 10.18653/v1/2025.findings-acl.1016.
- [21] X. Fang *et al.*, "Large language models (LLMs) on tabular data: prediction, generation, and understanding - a survey," 2024, *arXiv:2402.17944*.
- [22] S. Yao *et al.*, "Tree of thoughts: deliberate problem solving with large language models," *37th International Conference on Neural Information Processing Systems*, 2023, pp. 11809–11822, doi: 10.52202/075280-0517.
- [23] M. Besta *et al.*, "Graph of thoughts: solving elaborate problems with large language models," *38th AAAI Conference on Artificial Intelligence*, vol. 38, no. 16, pp. 17682–17690, Mar. 2024, doi: 10.1609/aaai.v38i16.29720.
- [24] J. Wang, J. Wang, B. Athiwaratkun, C. Zhang, and J. Y. Zou, "Mixture-of-agents enhances large language model capabilities," in *International Conference on Learning Representations*, 2025, pp. 33944–33963.
- [25] S. Dadkhah, E. C. P. Neto, R. Ferreira, R. C. Molokwu, S. Sadeghi, and A. A. Ghorbani, "CICIoMT2024: a benchmark dataset for multi-protocol security assessment in IoMT," *Internet of Things*, vol. 28, Dec. 2024, doi: 10.1016/j.iot.2024.101351.

BIOGRAPHIES OF AUTHORS



Mohamed Bal-Ghaoui    received his Ph.D. in AI from ENSIAS, University of Mohammed V in Rabat, Morocco. He is currently a project manager in Department of Research and Development at Audensiel in France, leading research in artificial intelligence and cybersecurity. His work focuses on developing and integrating artificial intelligence-driven solutions. He can be contacted at email: m.bal-ghaoui@audensiel.com.



Fayssal Sabri    is a double-degree engineer from Centrale Lyon and ENSAM Mekkès, specializing in artificial intelligence and data science. His research interests include ML, predictive modeling, optimization methods, and the development of intelligent systems for addressing complex real-world challenges. He can be contacted at email: fayssal.sabri@etu.ec-lyon.fr.