Intelligent Optimization in Massive Search Spaces: Evolutionary Learning and Reinforcement Selection

Xubin Wang

Dec 9, 2025

Motivation & Overview

- **The Challenge:** Modern AI faces enormous search spaces
  - **Example 1:** A dataset with 1000 features has 2^{1000} possible feature combinations
  - **Example 2:** Selecting 5 examples from 100 candidates for an LLM has 75 million possibilities
  - Traditional exhaustive search is computationally infeasible
- **Our Approach:** Two intelligent optimization frameworks
  - **MEL (Multi-Task Evolutionary Learning):** Automatically selects the most informative features from high-dimensional data using evolutionary algorithms
  - **RDES (Reinforcement Demonstration Selection):** Intelligently chooses the best examples to show Large Language Models (like ChatGPT) for improved performance
- **Key Insight:** Both use adaptive learning to find optimal solutions without exhaustive search

Why This Matters: Real-World Impact

Problem 1: Data Overload

- Medical diagnosis: Gene expression data with 20,000+ features
- Only 50-100 patient samples
- Challenge: Which genes are truly predictive?
- Traditional methods fail due to "curse of dimensionality"

Problem 2: LLM Effectiveness

- ChatGPT can solve tasks with examples (few-shot learning)
- Performance varies dramatically based on which examples you show
- Challenge: How to select the best examples automatically?
- Wrong examples $\rightarrow$ poor predictions

Our Goal: Smart algorithms that learn which choices work best

The Common Thread: Intelligent Search

- **What connects these problems?** Both require making smart choices from huge possibility spaces
- **Key Challenge:** Balancing competing objectives
  - **MEL:** Select *few but informative* features (compact yet accurate)
  - **RDES:** Select *similar but diverse* examples (relevant yet generalizable)
- **Our Solution:** Adaptive learning algorithms
  - **MEL** uses Evolutionary Computation: Population of candidate solutions evolve over time
  - **RDES** uses Reinforcement Learning: Agent learns from trial-and-error which examples work
- **Result:** Both methods automatically discover optimal strategies without human intervention

Presentation Outline

MEL: Efficient Multi-Task Evolutionary Learning for High-Dimensional Feature Selection

Xubin Wang, Haojiong Shangguan, Fengyi Huang, Shangrui Wu, and Weijia Jia

IEEE Transactions on Knowledge and Data Engineering (TKDE)

Front 1: High-Dimensional Optimization (MEL)

- **Context:** Feature selection is critical for enhancing model performance and reducing data dimensionality.
- **Challenge:** The increasing dimensionality of collected data leads to the "curse of dimensionality".
- This curse causes several problems: model overfitting, high model complexity, and lengthy training times.
- Feature selection is an **NP-hard problem**.
- Evolutionary Computation (EC) algorithms, though popular, struggle due to the vast search space and high computational complexity.

MEL Goal: Tackling NP-Hard Feature Selection

- The goal is to identify a compact feature subset that maximizes classification accuracy while reducing dimensionality.
- Traditional EC methods often underutilize and fail to share information effectively across related optimization sub-tasks.
- **MEL's Solution:** Integrate PSO with Multi-Task Learning (MTL) to jointly optimize related tasks and transfer knowledge efficiently.

MEL Focus: Efficiency and Effectiveness

- MEL's contributions are summarized by the "three E's":
- **Easy:** Simple yet effective PSO-based Multi-task Evolutionary Learning.
- **Effective:** Significant improvement in classification performance and compact feature subset selection.
- **Efficient:** Competitive running times, performing faster than standard PSO on high-dimensional problems.

The Feature Selection Problem

- Feature selection aims to find an optimal subset from n-dimensional features.
- This results in $2^n - 1$ possible feature combinations, making it a combinatorial optimization problem.
- Methods generally fall into three categories:
  1. **Filter Methods:** Evaluate features based on intrinsic data properties (e.g., variance, correlation). They are fast but ignore feature interaction with the model.
  2. **Wrapper Methods:** Treat selection as a search problem and evaluate subsets using a predictive model's accuracy. Often use stochastic search like PSO or GA.
  3. **Embedded Methods:** Perform selection during model training by assigning feature importance weights (e.g., regularization terms).

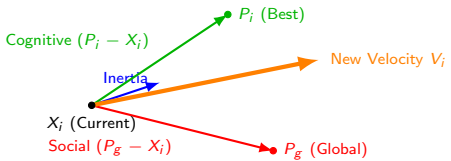
Particle Swarm Optimization (PSO) Background

- PSO mimics the social behavior of flocks searching for food.
- Each particle i in a $\mathcal{D}$-dimensional search space is characterized by its position $\vec{\mathcal{X}}_i$, velocity $\vec{\mathcal{V}}_i$, and individual best position $\vec{\mathcal{P}}_i$.
- The population tracks the global optimal position $\vec{\mathcal{P}}_g$.

Standard PSO Velocity Update

$$\vec{\mathcal{V}}_i^k = \omega \vec{\mathcal{V}}_i^{k-1} + c_1 r_1 (\vec{\mathcal{P}}_i^{k-1} - \vec{\mathcal{X}}_i^{k-1}) + c_2 r_2 (\vec{\mathcal{P}}_g^{k-1} - \vec{\mathcal{X}}_i^{k-1}) \tag{1}$$

- ω: inertia weight, influencing the degree of self-influence.
- c_1, c_2: learning factors for individual experience and population experience, respectively.
- r_1, r_2: random values in $[0, 1]$.



Standard PSO Position Update

$$\vec{\mathcal{X}}_i^k = \vec{\mathcal{X}}_i^{k-1} + \vec{\mathcal{V}}_i^{k-1} \tag{2}$$

- PSO is popular due to its simplicity and ease of implementation.
- **Limitation:** Directly applying PSO to high-dimensional datasets becomes inefficient due to the vast search space.

Multi-Task Learning (MTL) Definition

Definition (Multi-task Learning)

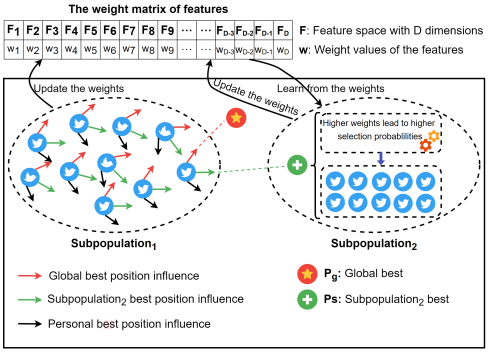
A machine learning method based on shared representation, which learns n related tasks $\{\mathcal{T}_i\}_{i=1}^n$ together and uses the association information between tasks $\{\mathcal{T}_i\}_{i=1}^n$ to improve generalization.

- MTL focuses on improving all tasks equally through shared information.
- **Analogy:** Learning to play ping pong helps accelerate learning tennis, and both skills help badminton (common skills improve performance in each).

MTL in Evolutionary Computation

- Integrating MTL with EC has shown promise for improved performance.
- **Key Advantages for PSO/EC in High Dimensions:**
  1. Maintains population diversity (via interacting sub-swarms) to avoid premature convergence.
  2. Guides particles towards generalizable features consistent across tasks, mitigating overfitting.
  3. Accelerates convergence by sharing informative samples.

MEL: Schematic Overview



The MEL method divides the parent population into $\vec{Sub}_1$ and $\vec{Sub}_2$.

- Both subpopulations learn feature importance during evolution.
- $\vec{Sub}_1$'s search is influenced by $\vec{Sub}_2$'s best solution (knowledge transfer).
- $\vec{Sub}_2$ searches based on the learned feature importance, prioritizing features with higher weights.

MEL Core Idea: Dual Subpopulations

- The parent population is divided equally and randomly into two subpopulations: $\vec{Sub}_1$ and $\vec{Sub}_2$.
- This dual population approach helps maintain diversity and balances exploration and exploitation.
- Since the population size (NP) is typically small (set to 20 in experiments), using only two subpopulations is deemed appropriate.

Algorithm 1: Initialization

Initialize a PSO population with NP individuals;
Initialize the weights of all features to 0;
Divide the population into two equal-sized subpopulations $\vec{Sub}_1$ and $\vec{Sub}_2$ that each perform a task;
Evaluate each particle, the individual best $\vec{\mathcal{P}}_i$, subpopulation best $\vec{\mathcal{P}}_s$ and global best $\vec{\mathcal{P}}_g$ are recorded;

- Particles are initialized randomly as real numbers.
- A threshold θ (set to 0.6) is used for binarization to determine selected features for evaluation.
- Feature selection is typically evaluated using a KNN classifier (K=3) via five-fold cross-validation.

Question 1: How to Identify Valuable Features?

- High-dimensional data is complex, containing relevant, irrelevant, and redundant features.
- In wrapper methods, direct measurement of individual feature importance is challenging because a learner evaluates feature **subsets**.
- **Solution:** Utilize historical information to learn feature importance over iterative evolutionary steps.

Definition 2: Feature Importance in MEL

Definition (Feature Importance)

The importance of a feature is measured by the change in accuracy resulting from its appearance or disappearance during the evolutionary process.

- This approach avoids preselection through thresholds (like ReliefF in prior work), ensuring original data information is retained.
- Feature weight $\mathcal{W}(\mathcal{F}_n)$ is updated based on the classification accuracy change: $\Delta acc = |acc_i - acc_{i-1}|$.

Knowledge Learning: Case 1 (Accuracy Improves)

- acc_i: Accuracy obtained in the current generation.
- acc_{i-1}: Accuracy obtained in the previous generation.
- **Case 1:** $acc_i > acc_{i-1}$ (Classification accuracy improves).

$$\mathcal{W}(\mathcal{F}_n) = \begin{cases} \mathcal{W}(\mathcal{F}_n) + (acc_i - acc_{i-1}), & \mathcal{F}_n \uparrow \\ \mathcal{W}(\mathcal{F}_n) - (acc_i - acc_{i-1}), & \mathcal{F}_n \downarrow \end{cases} \tag{3}$$

- $\mathcal{F}_n \uparrow$: Newly selected feature; $\mathcal{F}_n \downarrow$: Discarded feature.
- Newly selected features gain weight; discarded features lose weight.

Knowledge Learning: Case 2 (Accuracy Decreases)

- **Case 2:** $acc_i < acc_{i-1}$ (Classification accuracy decreases).

$$\mathcal{W}(\mathcal{F}_n) = \begin{cases} \mathcal{W}(\mathcal{F}_n) - (acc_{i-1} - acc_i), & \mathcal{F}_n \uparrow \\ \mathcal{W}(\mathcal{F}_n) + (acc_{i-1} - acc_i), & \mathcal{F}_n \downarrow \end{cases} \tag{4}$$

- Newly selected features lose weight (they hurt performance); discarded features gain weight (their absence helped performance).
- Features that remained unchanged take no specific action.

Question 2: Maintaining Population Diversity

- Traditional EC algorithms risk getting trapped in local optima, limiting global search ability.
- Multi-population mechanisms (like MTL) enhance diversity and avoid premature convergence.
- MEL integrates MTL into a two-subpopulation mechanism to balance exploration and exploitation.

Task 1: Search with External Influence ($\vec{Sub}_1$)

Definition (Task 1)

Subpopulation$_1$ ($\vec{Sub}_1$) conducts the search for the optimal feature subset based on the influence of individual best, *subpopulation*$_2$ ($\vec{Sub}_2$) best, and global best.

- $\vec{Sub}_1$ incorporates knowledge from $\vec{Sub}_2$'s best solution ($\vec{\mathcal{P}}_s$) during its velocity update.
- This leverages collective knowledge for exploration.

Knowledge Transfer to $\vec{Sub}_1$: PSO Modification

$$\begin{aligned}\vec{\mathcal{V}}_i^k = \omega\vec{\mathcal{V}}_i^{k-1} + c_1 r_1(\vec{\mathcal{P}}_i^{k-1} - \vec{\mathcal{X}}_i^{k-1}) + c_2 r_2(\vec{\mathcal{P}}_g^{k-1} - \vec{\mathcal{X}}_i^{k-1}) \\ + c_3 r_3(\vec{\mathcal{P}}_s^{k-1} - \vec{\mathcal{X}}_i^{k-1})\end{aligned} \tag{5}$$

- The additional term $c_3 r_3(\vec{\mathcal{P}}_s^{k-1} - \vec{\mathcal{X}}_i^{k-1})$ introduces influence from the best individual ($\vec{\mathcal{P}}_s$) in $\vec{Sub}_2$.
- c_3 is the learning factor for inter-subpopulation knowledge transfer, and r_3 is a random value in $[0, 1]$.

Task 2: Importance-Guided Search ($\vec{Sub}_2$)

Definition (Task 2)

Subpopulation$_2$ ($\vec{Sub}_2$) searches the optimal feature subset based on the feature importance information learned by the two subpopulations ($\vec{Sub}_1$ and $\vec{Sub}_2$).

- This task focuses on **exploitation** of the learned feature weights ($\mathcal{W}(\mathcal{F}_n)$).
- It aims to reduce evaluation costs by excluding features deemed irrelevant or detrimental.

Knowledge Transfer to $\vec{Sub}_2$: Feature Exclusion

- The feature weight matrix, updated via equations (3) and (4), may contain weights that are < 0, $= 0$, or > 0.
- $\vec{Sub}_2$ excludes features where $\mathcal{W}(\mathcal{F}_n) \leq 0$.
- This exclusion helps $\vec{Sub}_2$ concentrate its search resources on the most important (positive weight) features, improving efficiency.

Knowledge Transfer to $\vec{Sub}_2$: Selection Probability

- First, calculate the sum of positive feature weights (δ):

$$\delta = \sum_{n=1}^{\mathcal{D}} \mathcal{W}(\mathcal{F}_n), \forall \mathcal{W}(\mathcal{F}_n) > 0 \tag{7}$$

- Features with $\mathcal{W}(\mathcal{F}_n) > 0$ are selected with probability ρ:

$$\rho = \frac{\mathcal{W}(\mathcal{F}_n)}{\delta}, \forall \mathcal{W}(\mathcal{F}_n) \geq 0 \tag{8}$$

- Features with higher learned weights have a greater probability of being selected.

Objective Function: Fitness Balancing

- The fitness function f must balance two objectives: maximizing classification accuracy and minimizing the feature subset size.
- **Goal 1:** Smallest feature subset to avoid computational cost and high model complexity.
- **Goal 2:** Adequate number of features to ensure high classification accuracy.

Fitness Function (Equation 9)

$$f = \alpha \cdot error\_rate + \beta \frac{num\_of\_SF}{|\mathcal{F}|} \tag{9}$$

- *error_rate*: Classification error rate.
- *num_of_SF*: Size of selected features; $|\mathcal{F}|$: Total number of features.
- α: Classification result control weight; β: Feature size control weight.
- Parameters used: $\alpha = 0.9$ and $\beta = 0.1$, prioritizing accuracy.
- Optimization is achieved by minimizing f.

Datasets for High-Dimensional Feature Selection

- Experiments were conducted on 22 high-dimensional datasets.
- **Set 1:** 12 high-dimensional genetic datasets.
  - Feature counts range from 2,000 up to 54,675 dimensions (Stroke dataset).
  - Samples are typically small (e.g., Adenoma: 36 samples, CNS: 60 samples).
- **Set 2:** 10 datasets with larger sample sizes (thousands of samples) from Text, Image, and Gene domains.
  - Example: USPS (9,298 samples, 256 features); Pancancer (4,759 samples, 20,486 features).

Baselines: 24 Evolutionary Computation Methods

- MEL was compared against a diverse set of 18 classic and 6 recently published EC methods.
- **Classic Methods**:
  - Swarm-based (e.g., ABC, ACO, PSO, MBO).
  - Nature-inspired (e.g., BAT, CS, FA, FPA).
  - Evolutionary (e.g., DE, GA).
  - Bio-stimulated (e.g., FOA, GWO, HHO, WOA).
  - Physics-based (e.g., SA, HS, GSA, MVO).
- **Recent Methods**: SaWDE, FWPSO, DENCA, VGS-MOEA, MTPSO, PSO-EMT.

Overall Performance Summary (High-D Genetic Data)

- MEL achieved the **best average performance** across 12 genetic datasets compared to all 24 baselines.
- **Accuracy:** MEL achieved the highest average classification accuracy.
- **Subset Size:** MEL ranked second best, following the FWPSO method.
- **Efficiency:** MEL performed competitively, slightly slower only than SaWDE and SA.

Comparison: Accuracy vs. Swarm-based Methods

Dataset	ABC	ACO	PSO	MBO	MEL (Ours)	Best
Adenoma	0.9750 ± 0.0000	**1.0000** ± 0.0000	0.9750 ± 0.0000	0.9750 ± 0.0000	0.9975 ± 0.0079	ACO
ALL3	0.8160 ± 0.0053	0.8142 ± 0.0120	0.8192 ± 0.0067	0.8296 ± 0.0076	**0.8528** ± 0.0137	MEL
ALL4	0.8376 ± 0.0074	0.8495 ± 0.0045	0.8433 ± 0.0120	0.8542 ± 0.0192	**0.9035** ± 0.0219	MEL
Colon	0.8969 ± 0.0083	0.9118 ± 0.0125	0.9049 ± 0.0141	0.9197 ± 0.0178	**0.9282** ± 0.0136	MEL
Average	0.9255 ± 0.0079	0.9226 ± 0.0068	0.9285 ± 0.0072	0.9353 ± 0.0085	**0.9477** ± 0.0113	MEL

Table: Selected Accuracy Comparison with Swarm-based Methods

- MEL achieved the **highest average classification accuracy** (0.9477).
- MEL was the best performer on 8 out of 12 datasets.

Comparison: Subset Size vs. Swarm-based Methods

Dataset	ABC (Mean)	ACO (Mean)	PSO (Mean)	MBO (Mean)	MEL (Ours, Mean)	Smallest
ALL3 (12,625 F)	5081.0	1426.2	5188.2	4780.6	**1.0**	MEL
ALL4 (12,625 F)	5015.9	1158.5	5376.9	4750.2	**34.1**	MEL
CNS (7,129 F)	2874.7	2481.4	3023.1	2666.4	**57.2**	MEL
DLBCL (7,129 F)	2858.0	3632.3	2911.0	2708.5	**2697.8**	MEL
Average	5242.0	2330.8	5329.5	5110.7	**758.2**	MEL

Table: Selected Subset Size Comparison with Swarm-based Methods

- MEL significantly reduces model complexity.
- MEL's average subset size (758.2 features) is only approximately **33%** of the second-ranked ACO algorithm (2330.8 features).

Efficiency Comparison: MEL vs. PSO and ACO

Dataset	ABC (Time, s)	ACO (Time, s)	PSO (Time, s)	MBO (Time, s)	MEL (Ours, Time, s)	Fastest
Adenoma	71.0 ± 1.8	1965.7 ± 38.0	85.4 ± 1.8	82.0 ± 3.4	**55.5** ± 1.9	MEL
ALL3	194.7 ± 4.8	4812.2 ± 1545.9	207.3 ± 8.3	192.4 ± 2.8	**62.9** ± 2.3	MEL
ALL4	145.7 ± 1.1	8200.0 ± 11224.5	166.8 ± 4.8	150.4 ± 1.7	**66.3** ± 5.6	MEL
Stroke	177.4 ± 1.1	–	294.9 ± 6.3	247.7 ± 0.8	**115.7** ± 1.3	MEL
Average	114.6 ± 1.5	4264.2 ± 2756.5	138.9 ± 4.5	125.9 ± 1.7	**68.7** ± 3.9	MEL

Table: Selected Running Time Comparison with Swarm-based Methods

- MEL demonstrates the shortest average running time.
- MEL's running time is nearly **half** that of the original PSO algorithm.
- ACO is noted as unsuitable for high-dimensional data due to significantly longer runtimes.

Convergence Behavior: Accuracy

- Analysis of convergence curves (Figures 2, 4, 6, 8, 297) shows MEL's search capability.
- For swarm-based methods, MEL exhibits superior search capability across most datasets (e.g., ALL3, ALL4, CNS, Colon, Lymphoma).
- For nature-inspired methods, MEL exhibits slower initial convergence but demonstrates stronger global search ability, eventually outperforming others.
- For evolutionary algorithms (DE, GA), MEL shows significant advantages throughout the iterative process.

Convergence Behavior: Feature Subset Size

- Analysis of feature number convergence (Figures 3, 5, 7, 9, 300) highlights efficiency.
- ACO and MEL both demonstrate strong abilities to reduce relevant features.
- Unlike ACO, which shows fluctuation, MEL maintains better stability in feature reduction.
- MEL effectively searches for the optimal feature subset throughout most of the entire cycle, even when other methods stabilize quickly (Nature-inspired methods).

Comparison: MEL vs. Recently Published Methods

Dataset	SaWDE	FWPSO	DENCA	VGS-MOEA	PSO-EMT	MEL (Ours)	Best
Adenoma	0.9243	0.8361	0.9492	0.9960	0.9803	**1.0000**	MEL
ALL4	0.6977	0.8038	0.7703	0.8295	0.8670	**0.8846**	MEL
DLBCL	0.8990	0.8725	0.9354	0.9707	0.9685	**0.9874**	MEL
Prostate	0.7688	0.7955	0.8316	0.9065	**0.9272**	0.9092	PSO-EMT
Average	0.8508	0.8508	0.8765	0.9157	0.9213	**0.9455**	MEL

Table: Accuracy Comparison with State-of-the-Art EC Methods

- MEL achieves the **highest average accuracy** of 94.55%, outperforming these state-of-the-art algorithms.
- MEL obtains the best accuracy on 10 out of the 12 datasets.

Comparison: Subset Size

- FWPSO generally finds the smallest subsets (average size 2.8 features).
- However, FWPSO achieves a lower average accuracy (0.8508) compared to MEL (0.9455).
- MEL achieves a favorable trade-off, producing an average subset size of 666.2, prioritizing high accuracy (Effectiveness).

Comparison: Running Time

- SaWDE achieved the best average running time (5.9 seconds).
- MEL achieved a competitive average running time (48.9 seconds).
- Notably, methods like PSO-EMT incur extremely high computational costs (average 7596.5 seconds).
- MEL's efficiency validates the dual-task structure and importance-guided search in high-dimensional environments.

Performance on Large Sample Datasets

- Experiments were conducted on 10 datasets with thousands of samples to test scalability.
- MEL achieved the **highest average classification accuracy (88.64%)**.
- MEL outperformed others on 6 out of 10 large sample datasets.
- This validates MEL's effectiveness in selecting informative features even for data with thousands of samples and features.

Summary of MEL: Three E's Achieved

- MEL leverages Multi-task Learning to improve PSO's search efficiency in NP-hard feature selection.
- The dual-subpopulation structure ($\vec{Sub}_1$ for exploration, $\vec{Sub}_2$ for importance-guided exploitation) enhances robustness.
- **Result:** MEL achieves the best balance of high classification accuracy, optimal subset selection, and competitive running time among 24 EC algorithms.

Demonstration Selection for In-Context Learning via Reinforcement Learning

Xubin Wang, Jianfei Wu, Yichen Yuan, Deyu Cai, Mingzhe Li, and Weijia Jia

International Conference on Machine Learning (ICML) 2025

Front 2: Contextual Decision Making (RDES)

- **Context:** In-Context Learning (ICL) allows LLMs to perform tasks by providing a curated set of demonstrations as context, eliminating extensive retraining.
- **Key Challenge:** ICL effectiveness is critically dependent on selecting appropriate and representative demonstrations.
- **Limitation of Existing Methods:**
  - They often prioritize similarity, which risks **overfitting**.
  - They fail to dynamically adapt to the specific requirements of the reasoning task.

RDES Goal: Relevance and Diversity

- **Core Motivation:** Enhance performance by selecting demonstrations that maximize **relevance** while ensuring **diversity**.
- Similarity improves accuracy, but diversity promotes generalization.
- **RDES's Solution:** Frame demonstration selection as a sequential decision-making problem using RL.

Example: Diverse demonstrations (covering varied intent patterns) help the model avoid misclassifying edge cases, while similar-only examples may reinforce narrow predictions and lead to overfitting.

RDES Focus: Robustness and Generalization

- RDES dynamically selects demonstrations to enhance performance and model robustness for tasks amenable to ICL.
- The RL approach mitigates overfitting and improves generalization by balancing relevance and diversity.
- RDES integrates seamlessly with advanced reasoning techniques like Chain-of-Thought (CoT) prompting.

In-Context Learning (ICL) Overview

- LLMs perform tasks (e.g., text annotation, reasoning) by providing a small, curated set of demonstrations as context.
- ICL eliminates the need for extensive model retraining.
- This approach is particularly suitable for tasks with limited labeled data.

Prompt to LLM:

Instruction: Classify the sentiment of the text.

[Example 1] Text: "I love this movie!" $\rightarrow$ Label: Positive
[Example 2] Text: "The food was cold." $\rightarrow$ Label: Negative

[Test Input] Text: "The service was okay." $\rightarrow$ Label: **?**

$\Rightarrow$

LLM Output: Neutral

The Critical Role of Demonstration Selection

- The effectiveness of ICL is critically contingent upon selecting **appropriate and representative** demonstrations from the knowledge base.
- If overlooked, poor selection directly influences the model's ability to generalize and perform accurately in novel situations.
- Recent studies emphasize the need for effective selection strategies.

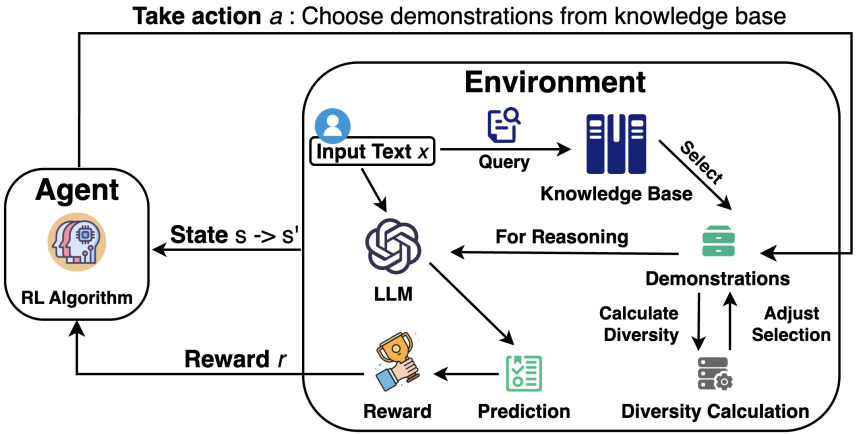
Challenge: Similarity Leads to Overfitting

- Traditional demonstration selection methods often prioritize only **similarity**.
- This oversight risks generating biased representations that do not generalize well to unseen data, ultimately hindering predictive accuracy.
- **Example:** If all demonstrations show only strongly positive sentiment, the model may misclassify a nuanced input.

The RDES Solution: Balancing Relevance and Diversity

- RDES aims to enhance performance by selecting demonstrations that maximize relevance while ensuring diversity.
- Diversity is critical for enhancing model generalization by enabling broader coverage of structures and concepts.
- This problem is framed as a **sequential decision-making problem**.
- RDES leverages Reinforcement Learning (RL) to dynamically balance exploration and exploitation in the selection process.

RDES Framework (Adaptive RL Approach)



RDES uses an RL Agent interacting with an Environment (Knowledge Base + LLM) to learn an optimal policy, guided by a reward with label-diversity.

RL Formulation: Sequential Decision Making

- RL provides a natural framework for sequential decision making in demonstration selection.
- The selection process is formalized as a finite-horizon Markov Decision Process (MDP) $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma)$.
- The policy learns to construct optimal demonstration sets through iterative trial-and-error interactions with the LLM.

MDP Component: State Space ($\mathcal{S}$)

- The state s_t captures the complete decision context by combining four key elements:
  1. **Textual Features:** Representation of the current input text (e.g., TF-IDF).
  2. **Demonstration Memory:** Embeddings of examples already selected.
  3. **Prediction History:** The LLM's previous predictions to track bias.
  4. **Diversity Tracking:** A score measuring the diversity of current labels.

MDP Component: Action Space ($\mathcal{A}$)

- The action space $\mathcal{A}$ is a discrete selection over candidate demonstrations $\mathcal{K}$ (the knowledge base).
- Action $a_t \in \{1, ..., |\mathcal{K}|\}$ indicates the chosen example index from the knowledge base.

MDP Component: Transition Dynamics ($\mathcal{P}$)

- The transition dynamics are deterministic, based on modifying the demonstration set.
- Taking action a_t (selecting candidate k_{a_t}) in state s_t leads to s_{t+1}.

$$s_{t+1} = f(s_t, a_t) = (x_t, E_t \cup \{k_{a_t}\}, \hat{y}_{t+1}, D_{t+1}) \tag{E.2}$$

- E_t is updated, and a new prediction $\hat{y}_{t+1}$ and diversity score D_{t+1} are generated.

MDP Component: Discount Factor (γ)

- The discount factor $\gamma \in [0, 1)$ is used.
- Emphasizing immediate rewards is suitable for the finite-horizon few-shot learning scenario, where a fixed number of examples (k) are selected.

The Dual Objective Reward ($\mathcal{R}$)

- The RL agent is guided by a multi-objective reward function that balances two critical goals:
  1. Maximizing prediction accuracy.
  2. Maximizing diversity gain.

$$\text{Reward} = \text{Accuracy} + \lambda \times \text{Diversity Gain}$$

- **Accuracy:** $+1$ if the LLM predicts correctly, 0 otherwise.
- **Diversity Gain:** Reward for adding a new label type to the context.
- λ: A dynamic weight that shifts focus from exploration (diversity) to exploitation (accuracy).

Balancing Tradeoff: The Diversity Coefficient (λ)

- The coefficient λ controls the exploration-exploitation tradeoff between maximizing immediate accuracy and seeking diversity.
- λ adapts during training via an annealing schedule.

$$\lambda(t) = \lambda_{\text{min}} + (\lambda_{\text{max}} - \lambda_{\text{min}})e^{-\eta t} \tag{E.4}$$

- This schedule prioritizes early exploration of diverse examples (high λ) before focusing on final accuracy (low λ).

RDES Optimization Strategy

- The RDES framework employs two primary RL algorithms based on the complexity of the state space:
  1. **Q-Learning** (RDES/B and RDES/C): Model-free, effective for discretizable state spaces.
  2. **PPO Variant** (RDES/PPO): Actor-critic, suited for high-dimensional state spaces.

RDES Optimization Strategy: Q-Learning

- **Method:** Q-Learning (Model-free RL).
- **Goal:** Learn a "Quality" table (Q-Table) that tells the agent: "In this situation, how good is it to pick this example?"
- **Process:**
  - The agent tries different examples (Exploration).
  - It receives rewards (Accuracy + Diversity).
  - It updates the Q-Table to remember the best choices for future reference.
- **Suitability:** Best for simpler tasks with discrete state representations.

RDES Optimization Strategy: PPO

- **Method:** Proximal Policy Optimization (PPO).
- **Why PPO?** For complex reasoning tasks, the state space is too huge for a table. We need Neural Networks.
- **Actor-Critic Architecture:**
  - **Actor (Policy Network):** Decides which example to pick next.
  - **Critic (Value Network):** Judges how good that decision was.
- **Key Feature:** PPO ensures stable learning by preventing the agent from making drastic changes to its policy too quickly ("Clipped Objective").

Unified Training Paradigm (Algorithm 1)

- Both Q-learning and PPO approaches share a unified training framework.
- The process involves iteratively:
  1. Sampling input x_i.
  2. Selecting initial demonstration candidates based on relevance, then applying diversity adjustment.
  3. Generating prompt p and obtaining prediction $\hat{y}$ from LLM $\mathcal{M}$.
  4. Computing diversity D and encoding state s.
  5. Selecting action a and calculating reward r (Accuracy + Diversity Improvement).
  6. Updating policy parameters θ using the chosen RL algorithm.

Step: Initial Demonstration Selection

Select demonstrations E with initial candidates based on relevance (e.g., top-k TF-IDF matches from $\mathcal{K}$) and apply diversity adjustment.

- The initial relevance selection sets the baseline context.
- The RL agent then determines the optimal selection sequence dynamically to maximize the dual objective (relevance/accuracy + diversity).

Standard Prompting Strategy

- The prompt is constructed by concatenating:
  1. The input text to be classified.
  2. The selected demonstrations (input-output pairs).
  3. The set of possible labels.
- The LLM then predicts the label based on this context.

Chain-of-Thought (CoT) Prompting Strategy (RDES/C)

- RDES/C integrates CoT reasoning into the prompt structure.
- Instead of just predicting the label, the LLM is asked to generate intermediate reasoning steps.
- This mimics human problem-solving and significantly improves performance on complex tasks.

RDES Datasets for Classification

- RDES was evaluated on four widely recognized intent classification and text classification datasets:
  1. **BANKING77:** Comprehensive set of 77 intents in the banking sector.
  2. **CLINC150:** 150 intents covering 10 distinct domains.
  3. **HWU64:** 64 intents covering 21 domains.
  4. **LIU54:** 54 intents covering 21 domains.
- Evaluation used a "challenge set sampling strategy" based on precision margin to ensure rigorous assessment.

RDES Datasets for Complex Reasoning

- To assess generalizability, RDES was tested on challenging benchmarks requiring complex reasoning:
  - **BigBenchHard (BBH):** Subsets including boolean expressions and web of lies.
  - **GSM-8K:** Mathematical reasoning problems.
  - **SST5:** Sentiment analysis task.
- For these supplementary tasks, RDES/PPO (PPO-based variant) was also evaluated.

LLMs Used in RDES Evaluation

- 14 LLMs were used, spanning closed-source and open-source models.
- **Closed-Source:** GPT-3.5-turbo, Doubao-lite-4k, Doubao-pro-4k, Hunyuan-lite.
- **Open-Source:** Gemma-2 (2B, 9B), LLaMA-3 (1B, 3B, 8B), Qwen-2.5 (7B, 14B), Qwen-1.5-72B.
- **Complex Reasoning Models:** Qwen-2.5-72B and DeepSeek-R1-32B.

Baseline Methods (Prompt Engineering)

- **Zero-Shot (ZS):** Tests generalization without prior demonstrations.
- **Knowledge Prompting (KP):** Provides contextual information for improved accuracy.
- **Least-to-Most (L2M):** Breaks tasks into manageable sequential steps.
- **Chain of Thought (CoT):** Encourages articulation of step-by-step reasoning.
- **Self-Refine (SF):** Iteratively critiques and refines its own solutions.

Baseline Methods (Demonstration Selection)

- **Few-Shot (FS):** Uses a limited number of randomly selected text-label pairs.
- **Few-Shot with CoT (FSC):** Combines FS demonstrations with explanations.
- **Active Demonstration Selection (AES):** Iteratively selects relevant demonstrations using RL (prior work).
- **Representative Demonstration Selection (RDS):** Identifies diverse subsets for better generalization (static diversity).
- **Adaptive ICL (ADA):** Optimization-free method focusing on uncertainty and semantic diversity.

Results: Reasoning Performance (Closed-Source)

Table: Performance comparison of methods designed to boost LLM reasoning across various datasets on closed-source LLMs, with a focus on accuracy.

Datasets	Models	Prompt Engineering Methods					Demonstration Selection Methods					Ours	
		ZS	KP	L2M	CoT	SF	FS	FSC	AES	RDS	ADA	RDES/B	RDES/C
BANKING77	GPT-3.5-turbo	0.340	0.240	0.260	0.200	0.380	0.520	0.320	0.260	0.240	0.360	0.767	**0.858**
	Doubao-lite-4k	0.300	0.300	0.300	0.320	0.300	0.500	0.360	0.300	0.280	0.400	0.750	**0.830**
	Doubao-pro-4k	0.500	0.400	0.500	0.480	0.600	0.540	0.540	0.700	0.680	**0.900**	0.838	0.888
	Hunyuan-lite	0.300	0.233	0.433	0.200	0.300	0.233	0.133	0.320	0.320	0.600	0.593	**0.775**
	Average	0.360	0.293	0.373	0.300	0.395	0.448	0.338	0.395	0.380	0.565	0.737	**0.838**
CLINC150	GPT-3.5-turbo	0.460	0.420	0.400	0.480	0.460	0.600	0.380	0.300	0.380	0.720	0.845	**0.949**
	Doubao-lite-4k	0.700	0.600	0.600	0.700	0.500	0.680	0.440	0.680	0.660	0.700	0.825	**0.927**
	Doubao-pro-4k	0.660	0.680	0.620	0.700	0.700	0.800	0.640	0.680	0.640	0.900	0.938	**0.961**
	Hunyuan-lite	0.633	**0.800**	0.767	0.700	0.633	0.467	0.500	0.480	0.620	**0.800**	0.730	0.772
	Average	0.613	0.625	0.597	0.645	0.573	0.637	0.490	0.535	0.575	0.780	0.835	**0.902**
HWU64	GPT-3.5-turbo	0.260	0.360	0.280	0.340	0.280	0.560	0.360	0.100	0.260	0.520	0.850	**0.914**
	Doubao-lite-4k	0.500	0.500	0.500	0.480	0.500	0.520	0.340	0.360	0.420	0.700	0.765	**0.873**
	Doubao-pro-4k	0.640	0.760	0.620	0.800	0.640	0.680	0.600	0.620	0.640	**1.000**	0.862	0.918
	Hunyuan-lite	0.533	0.367	0.333	0.433	0.233	0.600	0.433	0.540	0.320	0.700	0.514	**0.784**
	Average	0.483	0.497	0.433	0.513	0.413	0.590	0.433	0.405	0.410	0.730	0.748	**0.872**
LIU54	GPT-3.5-turbo	0.380	0.260	0.360	0.460	0.240	0.480	0.480	0.140	0.180	0.300	0.743	**0.868**
	Doubao-lite-4k	0.500	0.400	0.500	0.540	0.660	0.600	0.440	0.520	0.520	0.600	0.690	**0.841**
	Doubao-pro-4k	0.400	0.420	0.400	0.520	0.520	0.800	0.760	0.500	0.520	**0.900**	0.829	0.884
	Hunyuan-lite	0.533	0.500	0.567	0.700	0.633	0.367	0.500	0.460	0.620	0.560	0.565	**0.704**
	Average	0.453	0.395	0.457	0.555	0.513	0.562	0.545	0.405	0.460	0.590	0.707	**0.824**

Results: Reasoning Performance (Closed-Source) (cont.)

- RDES/B and RDES/C consistently outperform alternative methods across evaluated datasets (BANKING77, CLINC150, HWU64, LIU54).
- **RDES/C (with CoT) achieves highest accuracy in nearly all instances**.
- CoT and KP yield strong results, but task/dataset dependent.
- ADA and FSC competitive but generally outperformed by RDES/B and RDES/C.
- Doubao-pro-4k excelled, achieving peak performance of 0.961 on CLINC150 with RDES/C.
- GPT-3.5-turbo shows stable performance.
- RDES/C's incorporation of CoT consistently leads to superior performance.
- Underscores impact of advanced techniques, adaptive/CoT strategies are essential.

Results: Reasoning Performance (Open-Source)

Datasets	Models	Prompt Engineering Methods					Demonstration Selection Methods					Ours	
		ZS	KP	L2M	CoT	SF	FS	FSC	AES	RDS	ADA	RDES/B	RDES/C
BANKING77	Gemma-2-2B	0.200	0.280	0.200	0.200	0.260	0.300	0.340	0.280	0.220	**0.900**	0.831	0.861
	Gemma-2-9B	0.560	0.400	0.500	0.500	0.400	0.440	0.380	0.400	0.400	0.700	0.831	**0.886**
	LLaMA-3.2-1B	0.120	0.100	0.000	0.000	0.100	0.000	0.000	0.020	0.040	0.680	0.024	**0.744**
	LLaMA-3.2-3B	0.200	0.200	0.400	0.500	0.300	0.320	0.060	0.360	0.440	0.700	0.770	**0.805**
	LLaMA-3-8B	0.578	0.560	0.563	0.552	0.458	0.090	0.182	0.531	0.536	0.758	0.784	**0.847**
	Qwen-2.5-7B	0.700	0.480	0.600	0.420	0.480	0.440	0.180	0.440	0.420	0.700	0.803	**0.859**
	Qwen-2.5-14B	0.400	0.400	0.420	0.420	0.420	0.480	0.460	0.500	0.520	0.800	0.839	**0.868**
	Qwen-1.5-72B	0.529	0.480	0.524	0.528	0.551	0.653	0.612	0.509	0.542	0.775	0.785	**0.892**
	Average	0.411	0.363	0.401	0.390	0.371	0.340	0.277	0.380	0.390	0.752	0.708	**0.845**
CLINC150	Gemma-2-2B	0.400	0.600	0.420	0.460	0.560	0.560	0.540	0.500	0.380	0.800	0.875	**0.929**
	Gemma-2-9B	0.700	0.700	0.800	0.800	0.700	0.800	0.680	0.800	0.800	0.780	**0.864**	0.819
	LLaMA-3.2-1B	0.400	0.520	0.060	0.380	**0.600**	0.000	0.020	0.080	0.080	0.400	0.256	0.122
	LLaMA-3.2-3B	0.800	0.700	0.800	0.400	0.700	0.580	0.260	0.600	0.680	0.800	**0.845**	0.703
	LLaMA3-8B	0.523	0.439	0.594	0.504	0.569	0.007	0.285	0.571	0.543	0.767	**0.840**	0.783
	Qwen-2.5-7B	0.740	0.800	0.800	0.780	0.700	0.740	0.460	0.780	0.780	0.800	**0.879**	0.741
	Qwen-2.5-14B	0.900	0.900	0.900	0.800	0.840	0.700	0.700	0.740	0.740	0.900	**0.944**	0.792
	Qwen-1.5-72B	0.726	0.517	0.683	0.641	0.660	0.850	0.652	0.696	0.656	0.861	0.897	**0.963**
	Average	0.649	0.647	0.632	0.596	0.666	0.530	0.450	0.596	0.582	0.763	**0.800**	0.731
HWU64	Gemma2-2B	0.300	0.320	0.300	0.300	0.400	0.460	0.440	0.420	0.360	0.600	0.832	**0.851**
	Gemma2-9B	0.600	0.600	0.600	0.600	0.600	0.700	0.700	0.700	0.700	0.800	0.877	**0.910**
	LLaMA-3.2-1B	0.200	0.100	0.080	0.000	0.100	0.020	0.000	0.020	0.060	0.360	0.381	**0.687**
	LLaMA-3.2-3B	0.300	0.100	0.300	0.200	0.300	0.220	0.180	0.300	0.300	0.700	0.747	**0.817**
	LLaMA-3-8B	0.478	0.407	0.493	0.479	0.563	0.632	0.498	0.651	0.645	0.837	0.816	**0.859**
	Qwen-2.5-7B	0.780	0.700	0.800	0.600	0.800	0.640	0.540	0.760	0.740	0.800	0.805	**0.880**
	Qwen-2.5-14B	0.780	0.800	0.800	0.440	0.740	0.800	0.800	0.720	0.680	**0.900**	0.886	0.895
	Qwen-1.5-72B	0.698	0.615	0.676	0.661	0.668	0.825	0.817	0.749	0.774	0.877	0.867	**0.924**
	Average	0.517	0.455	0.506	0.410	0.521	0.537	0.497	0.540	0.532	0.734	0.776	**0.853**
LIU54	Gemma-2-2B	0.400	0.400	0.500	0.400	0.400	0.620	0.480	0.500	0.440	0.600	0.733	**0.854**
	Gemma-2-9B	0.500	0.500	0.600	0.600	0.600	0.580	0.580	0.500	0.500	**1.000**	0.722	0.837
	LLaMA-3.2-1B	0.200	0.160	0.300	0.400	0.080	0.040	0.040	0.360	0.320	**0.700**	0.058	0.651
	LLaMA-3.2-3B	0.400	0.400	0.400	0.300	0.400	0.360	0.320	0.500	0.400	0.600	**0.772**	0.749
	LLaMA-3-8B	0.358	0.409	0.428	0.360	0.392	0.396	0.320	0.347	0.312	0.763	0.779	**0.811**
	Qwen-2.5-7B	**0.800**	0.700	0.700	0.640	0.520	0.620	0.500	0.500	0.660	**0.800**	0.794	0.765
	Qwen-2.5-14B	0.700	0.860	0.700	0.660	0.700	0.660	0.600	0.740	0.780	**1.000**	0.849	0.743
	Qwen-1.5-72B	0.496	0.445	0.487	0.491	0.550	0.609	0.647	0.514	0.492	0.769	0.781	**0.880**
	Average	0.482	0.484	0.514	0.481	0.455	0.486	0.436	0.495	0.488	0.779	0.686	**0.786**

Results: Reasoning Performance (Open-Source)(cont.)

- Significant performance variations across datasets/models.
- **RDES/C consistently outperforms other methods in BANKING77** (average 0.845 vs ADA 0.752) and shows robustness in HWU64 (average 0.853 vs ADA 0.734).
- CLINC150 benefits from larger models (Qwen-1.5-72B).
- **On CLINC150, RDES/B (0.800) outperforms RDES/C (0.731)**, and ADA (0.763). This suggests dataset characteristics influence optimal RDES variant.
- ZS and KP show limitations compared to ADA and RDES.
- Larger models (Qwen-2.5-14B, Qwen-1.5-72B) show marked improvements, especially with RDES/C (synergistic effect of scale and technique).
- RDES methods, particularly with CoT, provide advantage across datasets.
- Dataset-specific trends underscore importance of tailored approaches.

Results on Complex Reasoning Tasks

- Experiments on BBH, GSM-8K, and SST5 validated RDES's effectiveness beyond simple classification.
- **DeepSeek-R1-32B:** RDES/C achieved the highest accuracy on BigBenchHard (web of lies: 1.00) and GSM-8K (0.73).
- **Qwen-2.5-72B:** RDES/PPO achieved competitive results, matching performance on BBH tasks (1.00) and GSM-8K (0.94).
- The success of RDES/C and RDES/PPO confirms that balancing relevance and diversity via RL maintains strong performance even in complex reasoning scenarios.

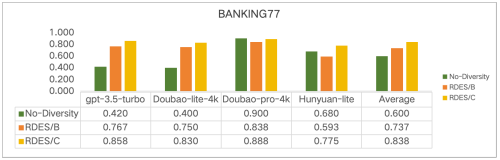
Impact of Varying Demonstrations (k)

- Experiments assessed performance when the number of demonstrations (k) was varied (3, 5, 7, 10) using Qwen-2.5-72B.
- **General Finding:** Performance of various methods, including RDES variants, changes based on the size of the demonstration set.
- For GSM-8K, RDES/PPO achieved a high score of 0.94 when $k = 3, 7$ and $k = 10$.
- These results show RDES's dynamic selection maintains effectiveness across different few-shot settings.

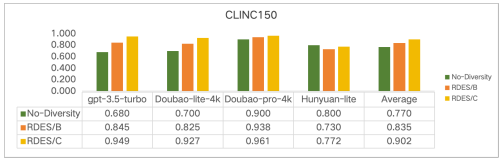
Results: Ablation Study on Diversity

- **Study Focus:** Impact of diversity mechanisms (No-Diversity, RDES/B, RDES/C) across closed-source and open-source models on four datasets.
- **Key Finding:** Incorporating diversity generally enhances model performance.
- **Closed-source Models (Figure 1):**
  - RDES/C consistently outperforms others across all datasets.
  - Example: BANKING77 avg accuracy: RDES/C (0.838) vs. No-Diversity (0.600).
- **Open-source Models (Figure 2):**
  - Performance varies by dataset.
  - BANKING77: RDES/C (0.845) vs. No-Diversity (0.747).
  - CLINC150: RDES/B (0.800) $>$ No-Diversity (0.768) and RDES/C (0.731).
  - HWU64: RDES/C (0.853) significantly boosts accuracy from No-Diversity (0.732).
  - LIU54: RDES/C (0.786) slightly higher than No-Diversity/RDES/B.
- **Conclusion:** A nuanced approach is needed for model and dataset pairing in open-source models.

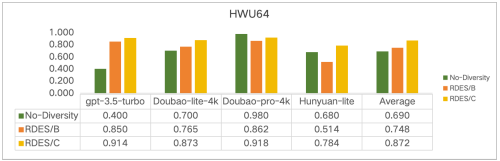
Results: Ablation Study on Diversity (Closed-source Models)



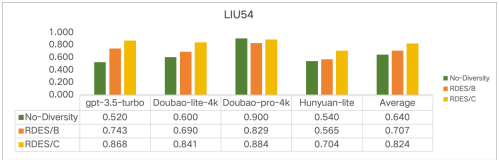
	gpt-3.5-turbo	Doubao-lite-4k	Doubao-pro-4k	Hunyuan-lite	Average
No-Diversity	0.420	0.400	0.900	0.680	0.600
RDES/B	0.767	0.750	0.838	0.593	0.737
RDES/C	0.858	0.830	0.888	0.775	0.838



	gpt-3.5-turbo	Doubao-lite-4k	Doubao-pro-4k	Hunyuan-lite	Average
No-Diversity	0.680	0.700	0.900	0.800	0.770
RDES/B	0.845	0.825	0.938	0.730	0.835
RDES/C	0.949	0.927	0.961	0.772	0.902



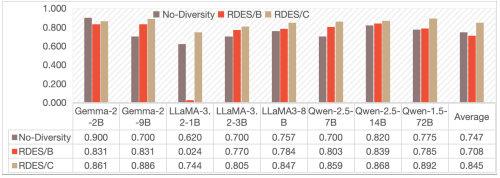
	gpt-3.5-turbo	Doubao-lite-4k	Doubao-pro-4k	Hunyuan-lite	Average
No-Diversity	0.400	0.700	0.980	0.680	0.690
RDES/B	0.850	0.765	0.862	0.514	0.748
RDES/C	0.914	0.873	0.918	0.784	0.872



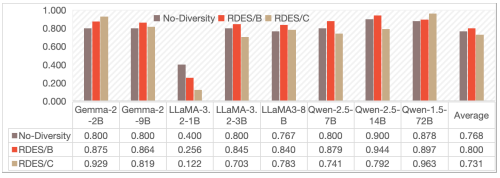
	gpt-3.5-turbo	Doubao-lite-4k	Doubao-pro-4k	Hunyuan-lite	Average
No-Diversity	0.520	0.600	0.900	0.540	0.640
RDES/B	0.743	0.690	0.829	0.565	0.707
RDES/C	0.868	0.841	0.884	0.704	0.824

Figure: Performance of various closed-source models across different datasets, highlighting the impact of diversity mechanisms.

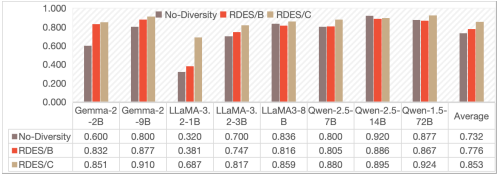
Results: Ablation Study on Diversity (Open-source Models)



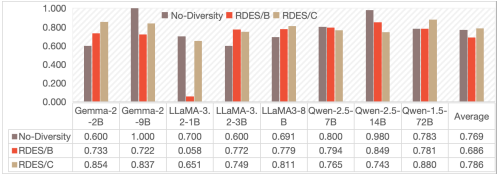
	Gemma-2-2B	Gemma-2-9B	LLaMA-3.2-1B	LLaMA-3.2-3B	LLaMA3-8B	Qwen-2.5-7B	Qwen-2.5-14B	Qwen-1.5-72B	Average
No-Diversity	0.900	0.700	0.620	0.700	0.757	0.700	0.820	0.775	0.747
RDES/B	0.831	0.831	0.024	0.770	0.784	0.803	0.839	0.785	0.708
RDES/C	0.861	0.886	0.744	0.805	0.847	0.859	0.868	0.892	0.845



	Gemma-2-2B	Gemma-2-9B	LLaMA-3.2-1B	LLaMA-3.2-3B	LLaMA3-8B	Qwen-2.5-7B	Qwen-2.5-14B	Qwen-1.5-72B	Average
No-Diversity	0.800	0.800	0.400	0.800	0.767	0.800	0.900	0.878	0.768
RDES/B	0.875	0.864	0.256	0.845	0.840	0.879	0.944	0.897	0.800
RDES/C	0.929	0.819	0.122	0.703	0.783	0.741	0.792	0.963	0.731



	Gemma-2-2B	Gemma-2-9B	LLaMA-3.2-1B	LLaMA-3.2-3B	LLaMA3-8B	Qwen-2.5-7B	Qwen-2.5-14B	Qwen-1.5-72B	Average
No-Diversity	0.600	0.800	0.320	0.700	0.836	0.800	0.920	0.877	0.732
RDES/B	0.832	0.877	0.381	0.747	0.816	0.805	0.886	0.867	0.776
RDES/C	0.851	0.910	0.687	0.817	0.859	0.880	0.895	0.924	0.853



	Gemma-2-2B	Gemma-2-9B	LLaMA-3.2-1B	LLaMA-3.2-3B	LLaMA3-8B	Qwen-2.5-7B	Qwen-2.5-14B	Qwen-1.5-72B	Average
No-Diversity	0.600	1.000	0.700	0.600	0.691	0.800	0.980	0.783	0.769
RDES/B	0.733	0.722	0.058	0.772	0.779	0.794	0.849	0.781	0.686
RDES/C	0.854	0.837	0.651	0.749	0.811	0.765	0.743	0.880	0.786

Figure: Performance of various open-source models across different datasets, highlighting the impact of diversity mechanisms.

Summary of RDES: Generalization and Robustness

- RDES leverages RL to optimize demonstration selection dynamically, addressing the limitations of static, similarity-based approaches.
- The dual objective reward explicitly drives the policy to balance prediction accuracy (relevance) and diversity gain (generalization).
- Integration with CoT (RDES/C) further boosts performance, particularly on complex tasks and closed-source models.

MEL vs RDES: Technical Comparison

Aspect	MEL	RDES
Optimization Target	Feature subset selection (2^d combinations)	Demonstration set selection (relevance + diversity)
Search Engine	Multi-task PSO (population-based)	Reinforcement Learning (Q-learning / PPO)
Learning Signal	Feature importance via temporal credit assignment: $w_j \leftarrow w_j + \Delta\text{Acc}$	Reward = Accuracy + λ(Diversity gain); annealed $\lambda(t)$
Update Mechanism	PSO velocity update with cross-swarm influence; probabilistic sampling $\propto \exp(w_j)$	Q-learning: TD update; PPO: clipped policy gradient with advantage A_t
Exploration-Exploitation	Two sub-swarms: $\vec{Sub}_1$ explores, $\vec{Sub}_2$ exploits via learned weights	ϵ-greedy (Q) or entropy bonus (PPO); λ annealing balances diversity vs accuracy
Convergence	PSO dynamics + knowledge transfer stabilizes; typically 100 iterations	Policy converges via gradient descent (PPO) or Q-value stabilization (Q-learning)
Computational Cost	$O(T \cdot NP \cdot d \cdot C_{\text{eval}})$; faster than standard PSO via guided search	$O(T \cdot \lvert K\rvert \cdot C_{\text{LLM}})$; dominated by LLM inference cost

MEL and RDES: The Unifying Challenge

- Both frameworks address the core problem: efficient and effective decision-making in vast search spaces.
- **MEL Search Space:** Feature combinations (2^n) in high dimensions.
- **RDES Search Space:** Demonstration subsets (combinatorial selection from knowledge base $\mathcal{K}$).
- Both utilize adaptive learning paradigms (EC or RL) rather than fixed heuristics.

Optimization Balance 1: Exploration vs. Exploitation

- **MEL's Approach (EC/MTL):** Explicit population division and interaction.
  - $\vec{Sub}_1$: Focuses on exploration using PSO with external guidance ($\vec{\mathcal{P}}_s$).
  - $\vec{Sub}_2$: Focuses on exploitation guided by learned feature importance.
- **RDES's Approach (RL/MDP):** Managed through the policy and reward structure.
  - λ-annealing schedule dynamically controls the balance, prioritizing early diversity (exploration).
  - ϵ-greedy strategy in Q-learning ensures continuous exploration.

Optimization Balance 2: Relevance vs. Diversity

- This dual focus is crucial for generalization in both domains.
- **MEL Relevance/Compactness:** The fitness function explicitly weights *error_rate* ($\alpha = 0.9$) and feature subset size ($\beta = 0.1$).
- **MEL Diversity:** Maintained by the two distinctive subpopulations and knowledge transfer to avoid local optima (premature convergence).

Optimization Balance 3: Relevance vs. Diversity (Cont.)

- **RDES Relevance (Accuracy):** Directly encoded as the primary reward $\mathbb{I}(y_{\text{true}} = \hat{y}_t)$.
- **RDES Diversity (Generalization):** Directly measured via Normalized Label Diversity D_t and rewarded as $D_{t+1} - D_t$.
- The dynamic, quantifiable reward signal allows RDES to adapt selection based on the current context deficit.

Adaptive Learning: Importance Guidance

- Both approaches feature mechanisms to learn and exploit importance dynamically.
- **MEL's Feature Importance:** Measures the effect of a feature's appearance/disappearance on classification accuracy over iterations. This learned weight directly guides $\vec{Sub}_2$'s search via selection probability ρ.
- **RDES's Contextual Importance:** The MDP state includes $\phi_x(x_t)$ (input relevance) and $\phi_y(\hat{y}_t)$ (prediction history), allowing the RL agent to learn which actions (demonstrations) lead to high reward for the current query.

The Power of Adaptive Meta-Heuristics

- **EC (MEL):** Robust search algorithms (PSO) enhanced by knowledge sharing (MTL). Suitable for continuous optimization spaces adapted for discrete problems (feature selection).
- **RL (RDES):** Dynamic policy learning tailored for sequential decision-making problems. Crucial for interacting with the complex, non-linear system represented by the LLM.
- Both show that performance improvement comes not from brute force search, but from *intelligent* guidance derived from task experience.

Analogy: The Expert Prospector

- Imagine an expert prospector searching for valuable resources (optimal features/demonstrations).
- **MEL's Approach** is like combining rugged seismic surveying (PSO search) with peer consultation (MTL knowledge transfer) to efficiently map vast mineral deposits (high-dimensional space) and find the most compact, high-yield veins (feature subset).
- **RDES's Approach** is like an adaptive sampling drone (RL agent) that learns to select samples based not only on proximity to the target (relevance) but also on ensuring the samples represent the full geological diversity of the area (generalization).

The Transition: From Features to Context

- The principles of optimization remain constant, even as the target shifts from classical ML features to LLM prompts.
- MEL showed us how to manage physical data complexity (curse of dimensionality).
- RDES shows us how to manage cognitive complexity (in-context reasoning) and maximize model utility without changing model weights.

Conclusion: MEL Achievements

- MEL proposed a simple yet effective PSO-based Multi-task Evolutionary Learning approach for feature selection.
- It effectively handles the curse of dimensionality by leveraging cooperative sub-populations and feature importance learning.
- MEL demonstrated superior classification accuracy and efficiency compared to 24 established EC methods across diverse high-dimensional datasets.

Conclusion: RDES Achievements

- RDES introduced a novel RL framework (Q-learning, PPO variant) for dynamically optimizing ICL demonstration selection.
- By explicitly balancing relevance and diversity through a dual-objective reward, RDES significantly mitigates overfitting and enhances generalization.
- RDES/C (with CoT) demonstrated superior performance across 14 LLMs on multiple classification and complex reasoning benchmarks.

Future Work: Evolutionary Learning (MEL)

- **Class Imbalance:** Future work should address class imbalance data, which is common in real-world scenarios, to further improve robustness.
- **Adaptive Parameters:** Explore making parameters (like ω, c_1, c_2, c_3) dynamic or self-adjusting based on population state.
- **Alternative Classifiers:** Evaluate MEL's effectiveness with different classification models beyond KNN.

Future Work: Reinforcement Selection (RDES) I

- **Refining Metrics:** Refine the diversity metrics used in the reward function to capture nuances beyond label distribution.
- **Adaptive CoT:** Explore making the utilization of Chain-of-Thought (CoT) adaptive within the RL framework.
- **Computational Efficiency:** Analyze and improve the computational cost and sample efficiency of the RL training process, which currently involves numerous LLM calls.

Future Work: Reinforcement Selection (RDES) II

- **Extension to Generative Tasks:** Extend RDES beyond classification and reasoning tasks to complex generative applications (e.g., summary, dialogue).
- **Alternative Retrieval:** Explore incorporating different initial retrieval methods beyond TF-IDF relevance.
- **Generalization Across Datasets:** Assess the transferability of the learned selection policies across different datasets or domains.

Thank You!

Questions?