SSEC: Semantic Self-Enhanced Classification for Few-Shot Tabular Data via Adversarial Reasoning

Min Li[1,2,3*], Yifei Zhang[4], Ming Yin[1,2,3], Neng Gao[1,3], Jia Peng[1,3*]
[1]Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
[2]School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China
[3]State Key Laboratory of Cyberspace Security Defense, Beijing, China
[4]Alibaba Group, Hangzhou, China
Emails: {minli, yinming, gaoneng, pengjia}@iie.ac.cn, zhanyi.zyf@alibaba-inc.com

***Abstract*—Few-shot classification of tabular data is challenging due to the limited labeled samples and unstable decision boundaries. Most existing approaches rely on statistical pattern matching or shallow metric learning, which often fail to capture the underlying semantic structure of tabular features, such as domain constraints and column meanings, under extreme data sparsity. As a result, models are prone to overfitting or boundary collapse in low-shot regimes. To address this problem, we propose SSEC (Semantic Self-Enhanced Few-Shot Classification), a learning method that explicitly incorporates semantic reasoning into few-shot tabular classification. SSEC constructs structured class-level semantic representations from limited labeled samples and iteratively refines them through an adversarial reasoning process, which sharpens decision boundaries and suppresses spurious correlations. Unlike purely statistical methods, SSEC leverages semantic constraints to provide a stronger inductive bias for few-shot learning. Experiments on multiple public tabular benchmarks demonstrate that SSEC performs competitively across varying few-shot settings, with particularly strong results in the 4-shot and 8-shot regimes. These results indicate that semantic enhancement is an effective strategy for improving robustness and generalization in few-shot tabular classification.**

***Index Terms*—few-shot learning, tabular data classification, large language models, adversarial reasoning, human-in-the-loop**

I. Introduction

In the era of the digital economy, data has solidified its status as a core strategic asset, underpinning intelligent governance and value creation across industries [1]. As organizations accelerate their digital transformation, the capacity to classify data accurately and efficiently becomes a prerequisite for data security, privacy compliance, and downstream analytical utility [2]. While data-driven paradigms have achieved remarkable success in scenarios abundant with labeled supervision, practical applications in scientific and industrial domains are frequently constrained by a "data scarcity" bottleneck.

Specifically, the explosion of business dimensions and the emergence of long-tail scenarios have made acquiring high-quality labeled samples prohibitively expensive or legally restricted [3]. This creates a pervasive "few-shot" constraint. Under such regimes, traditional machine learning algorithms (e.g., XGBoost) and deep tabular architectures (e.g., TabNet) often suffer from decision boundary collapse. Relying primarily on the statistical matching of low-level feature distributions, these models lack the inductive bias necessary to capture robust internal correlations in sparse feature spaces [4], [5].

To mitigate these limitations, recent research has pivoted towards Foundation Models. In the textual domain, Large Language Models (LLMs) have demonstrated exceptional few-shot generalization via In-Context Learning (ICL) [6]. Inspired by this, the tabular domain has witnessed the rise of Tabular Foundation Models, such as TabPFN [7] and its scalable successors, TabDPT [8] and Real-TabPFN [9]. While these models improve performance by pre-training on vast datasets, they predominantly operate on statistical patterns. They often fail to comprehend the underlying *semantic connotations* of metadata (e.g., column headers, domain-specific constraints), which human experts rely on heavily when classifying data with limited examples.

Conversely, while general-purpose LLMs possess the semantic reasoning capabilities lacking in tabular models, their direct deployment in vertical domains faces significant hurdles. In particular, LLMs often struggle to align unstructured language priors with the structured logic of tabular data, making it difficult to enforce domain-specific constraints (e.g., financial risk thresholds or clinical rules). Moreover, they are prone to cognitive bias and hallucinations when dealing with specialized domain logic, such as generating plausible but incorrect reasoning paths in tasks like credit assessment or medical triage [10], [11]. These limitations restrict their reliability and controllability in high-stakes tabular applications.

To bridge the gap between few-shot statistical constraints and high-level semantic reasoning, we propose Semantic Self-Enhanced Classification (SSEC). Unlike approaches that utilize LLMs merely as feature extractors or direct classifiers, SSEC leverages them as logic engines to distill general knowledge into robust, task-specific discriminative rules. We introduce a "Cognition-Competition-Feedback" closed-loop mechanism, which orchestrates a multi-agent adversarial game. This process iteratively refines categorical semantic profiles, effectively transforming vague general knowledge into precise decision boundaries. Furthermore, acknowledging the risks of automated reasoning, SSEC integrates a Human-in-the-Loop (HITL) adaptation mechanism. Drawing on recent insights into human-LLM interaction dynamics [12], our system fuses expert feedback with machine intelligence to ensure continuous,

reliable model evolution.

The main contributions of this paper are summarized as follows:

- We establish the SSEC framework, a novel paradigm that shifts from static descriptions to dynamic adversarial optimization, effectively filling the "semantic vacuum" in data-sparse scenarios.
- We construct a multi-agent adversarial loop that dynamically traces and resolves discriminative conflicts, significantly enhancing the robustness of semantic profiles against few-shot overfitting.
- We propose a Human-AI adaptation mechanism that supports online refinement and enables seamless integration of expert knowledge into the learning process, enhancing model adaptability in complex domain scenarios.

II. Related Work

A. Few-Shot Tabular Classification

Early deep tabular models (e.g., FT-Transformer [13]) often struggled to outperform GBDTs on small datasets due to the lack of inductive bias. To address this, the community has moved towards *Tabular Foundation Models*. TabPFN [7] pioneered this direction by introducing a Transformer pre-trained on synthetic priors to approximate Bayesian inference, enabling strong zero-shot performance.

Current research has evolved into two high-potential directions: scaling statistical laws and integrating semantic reasoning. On the statistical front, Ma et al. proposed TabDPT [8], which employs In-Context Learning (ICL) on massive real-world tabular corpora, demonstrating scaling capabilities analogous to LLMs. Similarly, Garg et al. introduced Real-TabPFN [9], showing that continued pre-training on curated real data significantly boosts robustness against distribution shifts. Parallel to statistical scaling, recent works have begun to exploit the reasoning capacity of LLMs for tabular tasks. Chain-of-Table [14] treats classification as a structured reasoning task, dynamically evolving table structures to guide LLM inference. Furthermore, TableLLM [15] and SheetAgent [16] investigate LLM-based approaches for table manipulation and reasoning in real-world spreadsheet scenarios. However, a critical gap persists: statistical foundation models largely remain "black boxes" that ignore rich metadata (e.g., column descriptions), while LLM-based reasoning methods often lack explicit optimization for discriminative boundaries in few-shot scenarios.

B. LLMs for Data Understanding and Reasoning

To inject semantic understanding into tabular tasks, recent works have begun to utilize LLMs directly. TabLLM [17] proposes a framework that serializes table rows into narrative texts, allowing LLMs to apply their world knowledge to tabular few-shot learning. TableLLM [15] extends this by training 7B+ parameter models specifically for manipulating tables in real-world office documents. Additionally, methods like Chain-of-Table [14] treat table classification as a structured reasoning task, dynamically evolving table structures to guide LLMs. Despite these advances, reliance on general LLMs introduces challenges: they incur high inference costs and can suffer from hallucinations when reasoning about specialized vertical domains without external verification [10], [11]. Our work addresses this by using LLMs to *generate logic* rather than just predicting labels.

C. Adversarial Reasoning and Human-in-the-Loop

Adversarial Agents: While adversarial training is well-established in GANs, its application to *logical reasoning* is a frontier research area. Liu et al. proposed the Generative Adversarial Reasoner (GAR) [18], which enhances LLM capabilities through a generator-discriminator game that refines chain-of-thought paths. Similarly, Chen et al. developed AgentCourt [19], where agent-lawyers engage in adversarial debate to refine legal arguments. SSEC adapts this "debate-for-truth" philosophy to optimize classification rules for tabular data.

Human-AI Collaboration: Real-world systems must adapt to concept drift. While Human-in-the-Loop (HITL) is standard, recent research scrutinizes the *quality* of interaction. Schroeder et al. [12] investigated LLM-assisted annotation, warning that humans might overly rely on machine suggestions. To counter this, SSEC employs an "active alignment" strategy [16], where experts do not merely label data but review and correct the *reasoning logic* generated by the model, creating a virtuous cycle of interpretability and accuracy.

III. Our Solution

A. Overall Framework

The SSEC framework is designed to tackle the boundary collapse caused by feature sparsity. Its core philosophy is the construction of a "Cognition-Competition-Feedback" self-enhancement loop. As illustrated in Fig. 1, the architecture consists of three deeply coupled stages:

- Heuristic Semantic Cognition: The process begins with an LLM acting as a general logic engine. Using our proposed Hierarchical Categorical Semantic Schema the model serializes raw structured samples and metadata (e.g., header semantics, value ranges). The LLM extracts not just statistical features but also leverages its prior knowledge to build an initial semantic profile covering core concepts, positive indicators, and boundary constraints.
- Adversarial Semantic Evolution: To eliminate potential biases in the initial semantics, we build an adversarial loop comprising a Generator, a Discriminator, an Optimizer, and Quality Assessor. The generator constructs challenging "hard negative" samples via feature masking, while the discriminator audits these samples based on current semantics, providing rational explanations. The optimizer then performs precise logical tracing to propose revisions, followed by a Quality Assessor that rigorously filters these candidates via multi-dimensional utility metrics to ensure robust version evolution.

- Human-in-the-Loop Adaptation: To handle concept drift, we introduce an adaptive evolution mechanism. During prediction, experts can perform real-time corrections, which are re-injected as "feedback patches" into the semantic profiles, enabling the model to improve continuously during deployment.

B. Category Semantic Initialization via Prompt Learning

The primary challenge in few-shot tasks is extrapolating discriminative categorical criteria from limited instances.To address this, we leverage prompt engineering to guide Large Language Models (LLMs) in synthesizing common feature patterns from sparse samples, thereby constructing a structured *Category Semantic Description*. We denote this foundational generation process as Category Semantic Initialization via Prompt Learning (CSI), which serves as the baseline module in our subsequent ablation study.

1) Categorical Semantic Schema Design: To ensure descriptions are highly structured, we define a three-layer Categorical Semantic Schema (CSS).

- Core Definition Layer ($\mathcal{C}$): Defines the macro-level ontological properties of the category. The LLM must perform a comprehensive scan of the input samples to summarize a "prototype definition" $\mathcal{C}_k$ for class C_k in the target domain. This prototype definition acts as the global semantic anchor for classification.
- Key Indicators Layer ($\mathcal{P}$) : This layer explicitly models the discriminative feature patterns. It includes the extraction of highly discriminative features and their interdependencies. The LLM identifies key indicators $\mathcal{P}_k = \{p_1, p_2, \ldots, p_n\}$ that reflect strong correlation with class membership, based on single-feature statistics and multivariate associations.
- Boundary Constraints Layer ($\mathcal{N}$): Aimed at establishing defensive classification boundaries by explicitly identifying critical differences between the current class and potential neighboring (hard negative) classes. These hard negative constraints $\mathcal{N}_k = \{n_1, n_2, \ldots, n_m\}$ effectively prevent semantic drift in overlapping feature regions by delineating where the class C_k should not extend.

2) Initial Mapping Function: To realize the transformation from concrete data to abstract logic, we construct a prompt-driven semantic mapping function. In an N-way K-shot task, the training dataset is defined as:

$$\mathcal{D}_{\text{train}} = \{(x_k^i, y_k) \mid i = 1, \ldots, K;\ k = 1, \ldots, N \tag{1}$$

Given the labeled support set for class c:

$$\mathcal{S}_c = \{x_c^1, x_c^2, \ldots, x_c^K\}, \quad \text{and corresponding metadata } \mathcal{M}$$

the initial semantic representation $\mathcal{D}_c^{(0)}$ is defined as:

$$\mathcal{D}_c^{(0)} = \text{Prompt}_{\text{init}}(\mathcal{M}, \text{Serialize}(\mathcal{S}_c), \text{Schema}) \tag{2}$$

Here, $\text{Prompt}_{\text{init}}$ serves as a heuristic induction operator that guides the LLM to extract deep structured knowledge in compliance with the predefined three-layer CSS structure.

Each structured instance x_c^i is first serialized into a natural language description. The LLM receives these serialized instances and, leveraging its internal inductive reasoning mechanisms, compares attribute differences across samples and outputs a highly condensed initial semantic sketch following the CSS:

$$\mathcal{D}_c^{(0)} = (\mathcal{C}_c, \mathcal{P}_c, \mathcal{N}_c) \tag{3}$$

This structured output format provides precise and atomically modifiable candidate components for the subsequent adversarial semantic optimization module.

C. Adversarial Semantic Evolution via Multi-Agent Collaboration(ASE)

To mitigate the cognitive bias embedded in the initial semantic descriptions under few-shot settings, we propose a closed-loop adversarial evolution framework driven by multiple collaborative agents. This framework is powered by a conflict-driven mechanism that iteratively refines category semantics through adversarial interactions.

1) Masking and Generation: We first employ a masking strategy on the serialized support samples to simulate uncertainty and generate challenging examples. For a given serialized instance $x \in \mathcal{S}_c$, represented as a sequence of semantic clauses $\{c_1, c_2, \ldots, c_L\}$, we independently apply a probabilistic masking process to each clause. Specifically, each clause c_j is transformed into a masked version $\tilde{c}_j^{\text{MASK}}$ as follows:

$$\tilde{c}_j^{\text{MASK}} = \begin{cases} [\text{MASK}], & \text{with probability } \rho \\ c_j, & \text{with probability } 1 - \rho \end{cases} \tag{4}$$

Applying this sampling process across all clauses yields a masked instance:

$$\tilde{x} = \{\tilde{c}_1, \tilde{c}_2, \ldots, \tilde{c}_L\} \tag{5}$$

A generator agent G then takes the masked input $\tilde{x}$, current semantic profile $\mathcal{D}_c^{(t)}$, and generation strategy as input to reconstruct the latent feature distribution:

$$x^{\text{syn}} \sim P_G(x \mid \tilde{x}, \mathcal{D}_c^{(t)}, \text{strategy}) \tag{6}$$

Unlike naive data augmentation, the generator operates under the semantic constraints of $\mathcal{D}_c^{(t)}$, targeting edge-case scenarios that challenge current classification boundaries—effectively creating high-utility "hard negatives."

2) Discrimination and Feedback: A discriminator agent D acts as a logical adjudicator, not only evaluating the fidelity of the synthetic sample but also diagnosing semantic flaws through cross-class reasoning. Formally, the discriminator maps a sample to a tuple:

$$D(x^{\text{syn}}) \to (v, \hat{y}, \mathcal{R}) \tag{7}$$

where $v \in \{0, 1\}$ denotes the fidelity score (i.e., whether the sample adheres to real-world distribution and domain logic), $\hat{y}$ is the predicted class label, and $\mathcal{R}$ is a rational explanation.

Two types of conflict signals can arise:

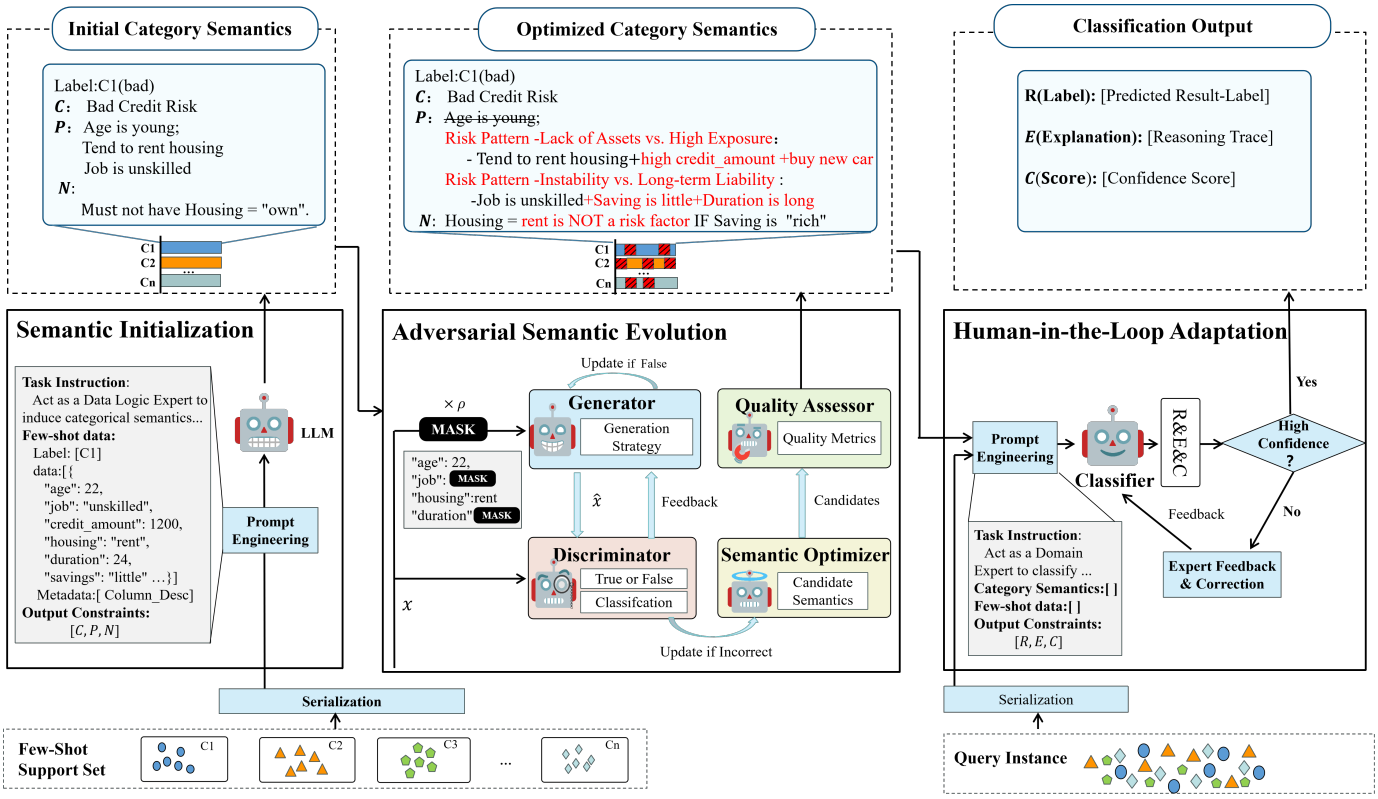

Fig. 1. Overview of the Proposed SSEC Framework.

- **Distributional Audit:** If $v < \tau$ (a predefined confidence threshold), the sample is flagged as distributionally implausible. $\mathcal{R}$ pinpoints violated attribute combinations, indicating a breakdown in the statistical alignment encoded in $\mathcal{P}_c$.
- **Semantic Confusion:** If $v \geq \tau$ but $\hat{y} \neq y_{\text{true}}$, semantic ambiguity is diagnosed. $\mathcal{R}$ then details the misclassification logic, often exposing boundary drift toward competitive categories.

This feedback mechanism converts non-differentiable logical contradictions into interpretable cognitive signals, offering precise modification targets and guiding semantic refinement toward greater exclusivity and robustness.

3) Conflict-Driven Semantic Evolution: Upon detecting conflicts, a semantic optimizer O is activated to trace the logic back to the root of failure within $\mathcal{D}_c^{(0)}$. It receives the full context tuple $(x^{\text{syn}}, \mathcal{D}_c^{(t)}, \mathcal{R})$ and proceeds with a hypothesis-evolution strategy:

1) **Defect Localization:** O parses $\mathcal{R}$ to identify failure layers (e.g., indicator-level $\mathcal{P}$ or constraint-level $\mathcal{N}$).
2) **Multi-Path Evolution:** It generates M candidate revisions $\mathcal{D}_c^{(t+1,m)} \in \mathcal{D}_{\text{cand}}$, covering strategies such as broadening, fine-grained adjustment, or constraint enhancement.
3) **Structured Reconstruction:** All candidates must comply with the CSS schema to ensure consistency and interpretability:

$$\mathcal{D}_{\text{cand}} = \{\mathcal{D}_c^{(t+1,m)}\}_{m=1}^{M} \tag{8}$$

This multi-path evolution balances generality and discrimination, transforming isolated conflict events into robust semantic corrections.

4) Multi-Dimensional Quality Evaluation: A quality assessor $\mathcal{E}$ then rigorously filters the candidate semantics. Each $\mathcal{D}_c^{(t+1,m)} \in \mathcal{D}_{\text{cand}}$ is evaluated across three criteria, forming a composite utility function U_D:

$$U_D = \lambda_1 \cdot \text{Sep} + \lambda_2 \cdot \text{Cov} - \lambda_3 \cdot \text{Comp} \tag{9}$$

Where:

- **Separability (Sep):** Measures semantic distance from neighboring class constraints $\mathcal{N}$.
- **Coverage (Cov):** Assesses alignment with support set $\mathcal{S}_c$, avoiding overfitting to hard negatives.
- **Compactness (Comp):** Penalizes structural complexity per Occam's razor.

Only when utility shows a significant positive delta is a candidate promoted. Otherwise, the optimizer rolls back to the last stable version. If improvements plateau or updates fall below a threshold ε, semantic convergence is declared.

This game-theoretic optimization and evaluation loop ensures that the evolved semantics are not only logically precise but also grounded in distributional fidelity and task-specific robustness.

D. Human-in-the-Loop Continual Optimization

To enhance long-term model robustness and ensure adaptive alignment with evolving business semantics, we introduce a human-guided continual optimization mechanism into the SSEC framework. Unlike static optimization routines, this mechanism enables real-time injection of expert knowledge during operational workflows, fostering a "getting-better-with-use" paradigm. Domain experts interact with the system by interpreting LLM-generated reasoning paths and supplying structured feedback, which continuously refines class-level semantic prototypes in response to emerging edge cases and evolving data patterns.

1) Semantic Inference-Driven Classification: Given a set of refined semantic descriptions $\{\mathcal{D}^*_{c_j}\}_{j=1}^{|\mathcal{C}|}$, each following the HCSS format $\mathcal{D}^*_{c_j} = \{C_j, P_j, N_j\}$, we serialize any test instance x into a natural language form $\text{Serialize}(x)$. This is incorporated into a prompt:

$$\Pi(x) = \text{Prompt}_{\text{cls}}\left(\text{Serialize}(x), \{\mathcal{D}^*_{c_1}, \ldots, \mathcal{D}^*_{c_{|\mathcal{C}|}}\}\right) \tag{10}$$

The LLM classifier outputs:

- A predicted label $\hat{y}$,
- A semantic reasoning trace $\mathcal{R}(x, \hat{y})$,
- A confidence score computed as:

$$\text{Conf}(x) = \frac{\exp(s_{\hat{y}})}{\sum_{j=1}^{|\mathcal{C}|}\exp(s_j)} \tag{11}$$

If $\text{Conf}(x) < \delta$, the instance is routed for human review.

2) Expert Feedback and Semantic Alignment: When flagged, the expert supplies a feedback tuple:

$$\mathcal{F}_x = (y^*, \mathcal{R}^*) \tag{12}$$

where y^* is the corrected label and $\mathcal{R}^*$ includes supporting logic or domain rules. If $\hat{y} \neq y^*$, the system performs a semantic correction:

$$\mathcal{D}^{t+1}_{y^*} = \mathcal{U}(\mathcal{D}^t_{y^*}, x, \mathcal{R}^*) \tag{13}$$

The update operator $\mathcal{U}$ refines $\mathcal{P}$, sharpens $\mathcal{N}$, and adjusts ambiguous concepts in C, enabling zero-retraining adaptation.

3) Prototype Evolution through Gold-Standard Instance Injection: To strengthen class semantics over time, expert-endorsed "gold-standard" samples are incrementally accumulated into a structured memory module $\mathcal{S}_{\text{expert}}$. These instances act as trusted reference anchors and support semantic stability during domain shifts. Each new gold sample x is integrated into its class prototype $\mathcal{C}_k$ using a momentum update mechanism:

$$\mathcal{C}_k^{\text{new}} \leftarrow (1-\alpha)\cdot\mathcal{C}_k^{\text{old}} + \alpha\cdot x \tag{14}$$

This continuous enrichment strategy enables adaptive expansion of the few-shot knowledge base while preserving semantic coherence, making the class representations increasingly robust in the face of data heterogeneity and label sparsity.

4) Semantic Refinement via Expert-Guided Prompt Injection: Beyond sample-level feedback, expert knowledge can be injected directly into the classification pipeline through adaptive prompt editing and semantic patching. When systematic misclassifications are detected, experts articulate high-level rationales $\mathcal{R}^*$ that are translated into structured constraints or new rules, which are then embedded into either:

- Updated prompt templates guiding future LLM inference, or
- The core semantic schema $\mathcal{D}^*_c$ for relevant categories.

This tight feedback loop allows the system to rapidly absorb tacit human knowledge and evolve its decision logic accordingly—without requiring full retraining or architecture modification.

By embedding real-time expert interventions into the model's semantic representation and inference mechanics, this mechanism enables classification systems to improve continually as they are used—creating a virtuous cycle of domain alignment, interpretability, and task robustness.

To facilitate reproducibility, the detailed prompt specifications for the main agents are presented in Fig. 2.

IV. Experiments

To comprehensively evaluate the effectiveness of the proposed **Semantic Self-Enhanced Classification (SSEC)** framework under data-scarce conditions, we conduct a series of few-shot classification experiments on a standard tabular benchmark. This section describes the dataset, experimental setup, baseline methods, and empirical results.

A. Dataset

We evaluate SSEC on two representative tabular benchmarks: Credit-g and Heart Disease, which typify data challenges in the financial and medical domains, respectively. These datasets are characterized by limited labeled samples, heterogeneous feature types, and ambiguous class boundaries—conditions that pose significant difficulty in few-shot scenarios.

Credit-g assesses credit risk based on diverse financial and personal attributes, reflecting real-world challenges in credit scoring where labeled data is often scarce. Heart Disease involves clinical risk prediction from compact and semantically entangled features, making it a rigorous testbed for evaluating model generalization and reasoning under minimal supervision.

B. Experimental Setup

a) Few-shot Evaluation Protocol: We adopt a standard few-shot evaluation protocol for tabular data. The full dataset is randomly split into a training pool and a held-out test set, where 20% of the data is reserved for testing. The remaining samples constitute the training pool used to construct few-shot support sets.

Under the N-way K-shot setting, we randomly sample K labeled instances per class from the training pool to form the support set, with $K \in 0, 4, 8, 16, 32$. Each experiment is

[SYSTEM ROLE]
Act as a Category Semantics Engineer. Your goal is to induce a structured semantic profile from few-shot tabular data.

[INPUT CONTEXT]
1. Label Set: {List of target classes}
2. Support Set (S): Small labeled training table with feature columns.
3. Metadata (M): Column names and type descriptions.

[TASK INSTRUCTION]
Analyze the support set to generate a 3-part Semantic Profile strictly following the schema:
1. Core Definition (Core):
  - Summarize the high-level semantic identity of the class.
2. Key Indicators (Pos):
  - Extract the Top-M (6-10) most discriminative features or patterns.
  - Focus on generalizable combinations rather than noise.
3. Boundary Constraints (Neg):
  - Identify exclusive constraints or counterexample cues.
  - Constraint: Be conservative; avoid absolute rejection unless certain.

[OUTPUT FORMAT]
Return a JSON object: { "core": "...", "pos": [...], "neg": [...] }

(a) Prompts for the Semantic Initialization Agent

[SYSTEM ROLE]
Act as an Adversarial Data Generator. Your goal is to reconstruct masked tabular instances to test the boundaries of the target class.
[INPUT CONTEXT]
1. Generation Strategy:
  - Construct "Hard Negative" samples.
  - The generated instance must statistically resemble the target class but lie near the decision boundary.
2. Target Semantics (D_t):
  - Indicators (P) and Constraints (N) of the target class.
3. Masked Input (x_tilde):
  - A serialized row with specific features replaced by [MASK].
[GENERATION CONSTRAINTS]
Fill in all [MASK] positions adhering to the following constraints:
1) Range Validity: Numeric values must be within reasonable domain bounds.
2) Logical Consistency: Imputed values must not contradict unmasked features.
3) Realism (Fidelity): The completed record must appear plausible.
4) Semantic Adherence: Maximize alignment with the Target Class Semantics.
[OUTPUT FORMAT]
Return strictly a JSON object containing the completed instance.

[GENERATION STRATEGY UPDATE]
Feedback:[Explanation of the Discriminator]
Updated Generation Strategy Only

(b) Prompts for the Generator

[SYSTEM ROLE]
Act as a Logical Adjudicator for tabular data. Your goal is to distinguish between real and plausible synthetic records, and to assign correct class labels.

[INPUT CONTEXT]
1. Candidate Classes:
  - Semantic Definitions (D) for all potential categories.
2. Evaluation Sample (x):
  - A tabular record (potentially synthetic/masked).

[DUAL-TASK INSTRUCTION]
1. Fidelity Audit (Authenticity Judgment):
  - Analyze if the record is logically consistent and plausible in the real world.
  - Output: Truth (Real/Fake) and a short rationale.
2. Semantic Classification (Category Judgment):
  - Assign the most appropriate class based on the Candidate Semantics.
  - Output: Predicted Category and a short rationale.

[OUTPUT FORMAT]
Strictly follow this schema:
- Truth: [Real | Fake] (Corresponds to fidelity score v)
- Truth Reason: [Rationale for plausibility]
- Category: [Class Name] (Corresponds to prediction y_hat)
- Category Reason: [Rationale R linking features to class semantics]

(c) Prompts for the Discriminator

[SYSTEM ROLE]
You are a Semantic Optimizer for few-shot tabular classificationRefine a target class's semantics using feedback from a discriminator.

[INPUT CONTEXT]
- current_description: class profile (core / pos / neg)
- feedback_summary: stable signals (min_count threshold)
- real_misclassified: true samples confused with others
- generated_true: approximate class prototypes

[TASK INSTRUCTION]
1. Be conservative under few-shot: few neg cues, avoid absolutes.
2. Prioritize stable signals; exclude rare or noisy patterns.
3. Keep core unchanged when possible; refine pos/neg only.
4. Repair confusion patterns via misclassified samples first.
5. Prefer comparative / soft phrasing over hard rules.

[OUTPUT FORMAT]
```json
{
 "candidates": [
   { "core": "...", "pos": [...], "neg": [...] },
   { "core": "...", "pos": [...], "neg": [...] }
 ]
}
```

(d) Prompts for the Semantic Optimizer

[SYSTEM ROLE]
You are a semantic quality evaluator for few-shot tabular classification. Evaluate each candidate profile for logical consistency and semantic utility.
[INPUT CONTEXT]
- base_core: Core definition of the target class.
- candidates: List of {core, pos, neg} structures.
[DIMENSIONAL ASSESSMENT]
Check:
1) flip — Core reverses risk direction?
2) contradiction — Internal conflicts: neg contradicts itself / core / pos?
3) Utility scores:
  - sep: separation from other classes
  - cov: coverage of support set
  - comp: compactness (lower = better)
[OUTPUT FORMAT]
"results": [
  {
   "candidate_index": 0,
   "flip": true/false, "flip_reason": "...",
   "contradiction": true/false,
   "contradiction_tag": "neg_internal_contradiction | neg_excludes_core | neg_excludes_pos | none",
   "contradiction_reason": "...",
   "sep": float, "cov": float, "comp": float
  }, ...]

(e) Prompts for the Quality Assessor

[SYSTEM ROLE]
Act as a Domain Expert. Perform classification by strictly aligning input samples with the provided Optimized Semantic Profiles.

[KNOWLEDGE BASE]
Candidate Classes & Optimized Semantics (D*):
{
 "Class_A": {
  "Core": "...",
  "Indicators": [P_A],
  "Constraints": [N_A]
 },
 "Class_B": { ... }
}
[INPUT DATA]
Samples to classify (Batch):{SAMPLES_BLOCK}

[OUTPUT SCHEMA]
Return a JSON object "results" where each element contains:
- "index": Sample ID.
- "prediction": The predicted class label (y_hat).
- "probs": Probability distribution across all classes (Used to compute Confidence Score).
- "reasoning": A brief trace explaining the semantic alignment.

(f) Prompts for the Inference Classifier

Fig. 2. Prompt Templates for the Core Agents of the Proposed Method.

TABLE I
PERFORMANCE COMPARISON UNDER DIFFERENT FEW-SHOT SETTINGS ON DIFFERENT DATASETS

Dataset	Method	0-shot	4-shot	8-shot	16-shot	32-shot
Credit-g	ProtoNet	—	0.54 ± 0.07	0.55 ± 0.07	0.60 ± 0.07	0.65 ± 0.05
	XGBoost	—	0.50 ± 0.00	0.51 ± 0.07	0.59 ± 0.05	0.66 ± 0.03
	TabPFN	—	0.58 ± 0.08	0.59 ± 0.03	$\mathbf{0.64 \pm 0.06}$	$\mathbf{0.69 \pm 0.07}$
	SSEC	$\mathbf{0.54 \pm 0.03}$	$\mathbf{0.60 \pm 0.09}$	$\mathbf{0.64 \pm 0.05}$	$\mathbf{0.64 \pm 0.04}$	$\mathbf{0.69 \pm 0.04}$
Heart	ProtoNet	—	0.81 ± 0.12	0.82 ± 0.09	0.87 ± 0.06	0.90 ± 0.06
	XGBoost	—	0.50 ± 0.00	0.55 ± 0.14	0.84 ± 0.07	0.88 ± 0.04
	TabPFN	—	0.84 ± 0.06	$\mathbf{0.88 \pm 0.05}$	0.87 ± 0.06	$\mathbf{0.91 \pm 0.02}$
	SSEC	$\mathbf{0.63 \pm 0.02}$	$\mathbf{0.87 \pm 0.04}$	$\mathbf{0.88 \pm 0.07}$	$\mathbf{0.88 \pm 0.04}$	0.90 ± 0.03

repeated with multiple random seeds (seed = 0, 1, 2, 3, 4), and the mean performance along with standard deviation is reported.

b) Evaluation Metric: We use the Area Under the Receiver Operating Characteristic Curve (AUC) as the primary evaluation metric. AUC is particularly suitable for few-shot tabular classification, as it evaluates ranking quality across all decision thresholds and is less sensitive to class imbalance than accuracy-based metrics. This makes AUC a more reliable indicator of decision boundary quality under data-scarce conditions.

c) Model Configuration: We employed OpenAI's GPT-5.1 (snapshot gpt-5.1-2025-11-13 via OpenRouter[1]) as the core engine, leveraging its adaptive reasoning capabilities. To balance stability and diversity, we set the temperature to $\tau = 0.2$ for deterministic reasoning tasks and $\tau = 0.7$ for adversarial generation.

C. Baseline Methods

We compare SSEC against several representative few-shot classification baselines, including:

- **ProtoNet** [20]: A metric-based few-shot learning method that classifies samples based on distances to class prototypes.
- **XGBoost** [21]: A gradient-boosted decision tree model representing strong traditional machine learning baselines for tabular data.
- **TabPFN** [7]: A transformer-based model pre-trained on synthetic tabular tasks for fast adaptation.

All baseline methods are evaluated under the same few-shot protocol and data splits to ensure fair comparison.

D. Results and Analysis

Table I presents the AUC scores across varying few-shot settings on both the *Credit-g* and *Heart* datasets. These results reflect the behavior of each method under limited supervision and highlight the impact of semantic modeling in SSEC.

On **Credit-g**, SSEC shows clear advantages in the low-shot regime. In the 0-shot setting, it achieves 0.54 ± 0.03, traditional baselines such as ProtoNet, XGBoost, and TabPFN are not applicable under 0-shot, as they require at least one labeled instance per class. This highlights SSEC's unique capability to induce coarse decision boundaries solely from semantic priors, without any labeled supervision.As the number of labeled examples increases, SSEC demonstrates steady improvements—reaching 0.64 ± 0.04 at 16-shot and 0.69 ± 0.04 at 32-shot—comparable to the strongest baselines, while maintaining low variance.

On **Heart**, SSEC consistently yields favorable outcomes across all few-shot levels. It achieves the best or tied-best results from 0- to 16-shot, and remains close to the top-performing model at 32-shot (0.90 ± 0.03 vs. 0.91 ± 0.02). These results suggest that the semantic structure and adversarial refinement in SSEC provide a stable inductive bias, particularly beneficial in low-data regimes.

Overall, SSEC demonstrates reliable generalization under varying degrees of supervision, without relying on large-scale training or heavy parametrization. While its advantage is most evident in low-resource settings, it remains competitive as supervision increases, supporting its suitability for real-world scenarios where labeled data is scarce or costly to obtain.

E. Ablation Study

To assess the contribution of each component in SSEC, we conducted an ablation study on the Credit-g dataset using three configurations: (1) **LLM-as-Classifier**, which serves as the baseline by directly prompting the LLM to output class labels without structured modeling; (2) **LLM + Semantic Initialization**, which incorporates class-level semantic prototypes; and (3) **Full SSEC: + Semantic Refinement**, which further adds the adversarial loop for iterative optimization.

As shown in Table II, introducing semantic initialization improves performance across most few-shot settings, indicating that even static class-level semantics provide a useful inductive bias. In particular, the 8-shot AUC increases from 0.61 to 0.63. The full SSEC framework yields further gains, notably at 4-shot and 16-shot, where the AUC reaches 0.60 and 0.64, respectively. These improvements suggest that adversarial refinement helps sharpen decision boundaries and mitigate overfitting.

[1]https://openrouter.ai/openai/gpt-5.1

TABLE II
ABLATION STUDY OF SSEC COMPONENTS ON CREDIT-G DATASET (AUC)

Configuration	4-shot	8-shot	16-shot	32-shot
LLM-as-Classifier	0.58 ± 0.08	0.61 ± 0.08	0.61 ± 0.05	0.69 ± 0.03
LLM + Semantic Initialization	0.59 ± 0.06	0.63 ± 0.07	0.62 ± 0.02	0.69 ± 0.06
Full SSEC: + Semantic Refinement	$\mathbf{0.60 \pm 0.09}$	$\mathbf{0.64 \pm 0.05}$	$\mathbf{0.64 \pm 0.04}$	$\mathbf{0.69 \pm 0.04}$

In the 32-shot regime, all methods converge to a similar AUC of around 0.69, reflecting the reduced marginal benefit of semantic modeling when sufficient supervision is available. This reinforces that SSEC is especially effective under data-scarce conditions (e.g., $\leq$16 shots), where structured semantic reasoning is most needed.

V. CONCLUSION

This paper proposes SSEC, a semantic self-enhanced classification framework that integrates large language models, adversarial optimization, and label-guided feedback for few-shot tabular learning. Unlike purely statistical methods, SSEC constructs and iteratively refines structured semantic profiles to improve the robustness of decision boundaries. Experiments on the Credit-g and Heart datasets demonstrate that SSEC performs favorably under low-shot supervision, with competitive performance as the number of labeled samples increases.

ACKNOWLEDGMENT

The authors acknowledge helpful feedback and technical suggestions received during the preparation of this manuscript. All content and revisions are the sole responsibility of the authors.

REFERENCES

[1] D. Zha, Z. P. Bhat, K.-H. Lai, F. Yang, Z. Jiang, S. Zhong, and X. Hu, "Data-centric artificial intelligence: A survey," *ACM Computing Surveys*, vol. 57, no. 5, pp. 1–42, 2025.

[2] N. Gupta, S. Mujumdar, H. Patel, S. Masuda, N. Panwar, S. Bandyopadhyay, S. Mehta, S. Guttula, S. Afzal, R. Sharma Mittal *et al.*, "Data quality for machine learning tasks," in *Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining*, 2021, pp. 4040–4041.

[3] Y. Wang, Q. Yao, J. T. Kwok, and L. M. Ni, "Generalizing from a few examples: A survey on few-shot learning," *ACM computing surveys (csur)*, vol. 53, no. 3, pp. 1–34, 2020.

[4] L. Grinsztajn, E. Oyallon, and G. Varoquaux, "Why do tree-based models still outperform deep learning on typical tabular data?" *Advances in neural information processing systems*, vol. 35, pp. 507–520, 2022.

[5] V. Borisov, T. Leemann, K. Seßler, J. Haug, M. Pawelczyk, and G. Kasneci, "Deep neural networks and tabular data: A survey," *IEEE transactions on neural networks and learning systems*, vol. 35, no. 6, pp. 7499–7519, 2022.

[6] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell *et al.*, "Language models are few-shot learners," *Advances in neural information processing systems*, vol. 33, pp. 1877–1901, 2020.

[7] N. Hollmann, S. Müller, K. Eggensperger, and F. Hutter, "Tabpfn: A transformer that solves small tabular classification problems in a second," *arXiv preprint arXiv:2207.01848*, 2022.

[8] J. Ma, V. Thomas, R. Hosseinzadeh, A. Labach, J. C. Cresswell, K. Golestan, G. Yu, A. L. Caterini, and M. Volkovs, "Tabdpt: Scaling tabular foundation models on real data," in *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

[9] A. Garg, M. Ali, N. Hollmann, L. Purucker, S. Müller, and F. Hutter, "Real-tabpfn: Improving tabular foundation models via continued pre-training with real-world data," *arXiv preprint arXiv:2507.03971*, 2025.

[10] Z. Ji, N. Lee, R. Frieske, T. Yu, D. Su, Y. Xu, E. Ishii, Y. J. Bang, A. Madotto, and P. Fung, "Survey of hallucination in natural language generation," *ACM computing surveys*, vol. 55, no. 12, pp. 1–38, 2023.

[11] L. Huang, W. Yu, W. Ma, W. Zhong, Z. Feng, H. Wang, Q. Chen, W. Peng, X. Feng, B. Qin *et al.*, "A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions," *ACM Transactions on Information Systems*, vol. 43, no. 2, pp. 1–55, 2025.

[12] H. Schroeder, D. Roy, and J. Kabbara, "Just put a human in the loop? investigating llm-assisted annotation for subjective tasks," in *Findings of the Association for Computational Linguistics: ACL 2025*, 2025, pp. 25 771–25 795.

[13] Y. Gorishniy, I. Rubachev, V. Khrulkov, and A. Babenko, "Revisiting deep learning models for tabular data," *Advances in neural information processing systems*, vol. 34, pp. 18 932–18 943, 2021.

[14] Z. Wang, H. Zhang, C.-L. Li, J. M. Eisenschlos, V. Perot, Z. Wang, L. Miculicich, Y. Fujii, J. Shang, C.-Y. Lee *et al.*, "Chain-of-table: Evolving tables in the reasoning chain for table understanding," *arXiv preprint arXiv:2401.04398*, 2024.

[15] X. Zhang, S. Luo, B. Zhang, Z. Ma, J. Zhang, Y. Li, G. Li, Z. Yao, K. Xu, J. Zhou *et al.*, "Tablellm: Enabling tabular data manipulation by llms in real office usage scenarios," in *Findings of the Association for Computational Linguistics: ACL 2025*, 2025, pp. 10 315–10 344.

[16] Y. Chen, Y. Yuan, Z. Zhang, Y. Zheng, J. Liu, F. Ni, and J. Hao, "Sheetagent: A generalist agent for spreadsheet reasoning and manipulation via large language models," in *ICML 2024 Workshop on LLMs and Cognition*, 2024.

[17] S. Hegselmann, A. Buendia, H. Lang, M. Agrawal, X. Jiang, and D. Sontag, "Tabllm: Few-shot classification of tabular data with large language models," in *International conference on artificial intelligence and statistics*. PMLR, 2023, pp. 5549–5581.

[18] Q. Liu, L. Ye, W. Ma, Y.-C. Chou, and A. Yuille, "Generative adversarial reasoner: Enhancing llm reasoning with adversarial reinforcement learning," *arXiv preprint arXiv:2512.16917*, 2025.

[19] G. Chen, L. Fan, Z. Gong, N. Xie, Z. Li, Z. Liu, C. Li, Q. Qu, H. Alinejad-Rokny, S. Ni *et al.*, "Agentcourt: Simulating court with adversarial evolvable lawyer agents," in *Findings of the Association for Computational Linguistics: ACL 2025*, 2025, pp. 5850–5865.

[20] J. Snell, K. Swersky, and R. Zemel, "Prototypical networks for few-shot learning," in *Advances in neural information processing systems*, vol. 30, 2017.

[21] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, 2016, pp. 785–794.