Unsupervised Feature Transformation via In-context Generation, Generator-critic LLM Agents, and Duet-play Teaming

Nanxu Gong[1] , **Xinyuan Wang**[1] , **Wangyang Ying**[1] , **Haoyue Bai**[1] , **Sixun Dong**[1] , **Haifeng Chen**[2] , **Yanjie Fu**[1] *

[1]School of Computing and Augmented Intelligence, Arizona State University
[2]NEC Laboratories America
{nanxugong, xwang735, wying4, haoyueba, sixundong, yanjie.fu}@asu.edu, haifeng@nec-labs.com

Abstract

Feature transformation involves generating a new set of features from the original dataset to enhance the data's utility. In certain domains like material performance screening, dimensionality is large and collecting labels is expensive and lengthy. It highly necessitates transforming feature spaces efficiently and without supervision to enhance data readiness and AI utility. However, existing methods fall short in efficient navigation of a vast space of feature combinations, and are mostly designed for supervised settings. To fill this gap, our unique perspective is to leverage a generator-critic duet-play teaming framework using LLM agents and in-context learning to derive pseudo-supervision from unsupervised data. The framework consists of three interconnected steps: (1) Critic agent diagnoses data to generate actionable advice, (2) Generator agent produces tokenized feature transformations guided by the critic's advice, and (3) Iterative refinement ensures continuous improvement through feedback between agents. The generator-critic framework can be generalized to human-agent collaborative generation, by replacing the critic agent with human experts. Extensive experiments demonstrate that the proposed framework outperforms even supervised baselines in feature transformation efficiency, robustness, and practical applicability across diverse datasets. Our code is publicly available at https://github.com/NanxuGong/LPFG.

1 Introduction

Feature transformation aims to rebuild a new feature space from an original feature set (e.g. $[f_1, f_2] \rightarrow [\frac{f_1}{f_2}, f_1 - f_2, \frac{f_1+f_2}{f_1}]$). Feature transformation can advance the power (structural, predictive, interaction, and expression levels) of data to make data AI-ready. In many practices, feature transformation is conducted either by human experts or by machine-assisted search guided through downstream task feedback. In certain domains, like material synthesis and performance screening, feature transformation is particularly useful in capturing interactions and compositions within material formulas to identify performance drivers. However, 1) there are millions of candidate material ingredients (i.e., features) for material synthesis; thus, it is inefficient to explore and search all feature combinations and interaction possibilities; 2) obtaining supervised material performance labels often requires time-intensive and costly in-lab experiments. This practical challenge highlights the need for a new AI task: efficient and unsupervised feature transformation (EUFT).

There are two major challenges in solving EUFT: 1) efficient transformation, and 2) unsupervised transformation. Firstly, there is an exponentially expanding range of feature combination possibilities in a feature space, leading to an overwhelmingly large discrete search space. Efficient transformation is to answer: how can we avoid searching a large feature combination space when generating feature transformations? Secondly, in supervised settings, most methods exploit predictive accuracy feedback of a transformed feature set on a downstream ML model to guide optimal feature transformation search. Under unsupervised settings, there is no supervised knowledge as guidance. Unsupervised transformation aims to answer: how can we discover supervision knowledge from unsupervised data to steer the optimal feature transformation generation?

There are significant gaps in current methodologies for EUFT. 1) *Manual feature transformation* requires domain and empirical expertise to formulate task-specific strategies, thus is inefficient and doesn't generalize well in unsupervised settings. 2) There are studies that solve feature transformation as *discrete or continuous search tasks* [Wang *et al.*, 2022; Wang *et al.*, 2023a]. Technical solutions include reinforcement learning, genetic algorithm, and generative learning based reformulations. They either search optimal feature sets in a discrete or continuous space. However, the models are time-consuming and require downstream supervised feedback to guide search. 3) *LLM agent based methods* [Gong *et al.*, 2024; Zhang *et al.*, 2024; Hollmann *et al.*, 2024] interpret the prompt using their pretrained general and world token knowledge to generate outputs that align with the given patterns, scores, or comparison patterns of features to generate tokenized feature transformations. But, existing methods target at supervised settings, instead of unsupervised setting. They regard LLM as a generator and ignore its other abilities: in-context learning enabled teaming, diagnosis and

[*] Corresponding author

critics, duet-play can deliver better feature transformations in more challenging computational or learning settings.

Our insights: a duet-play generator-critic teaming perspective to derive supervision from unsupervised data. We highlight two research insights. To address efficient transformation, we show that we can tokenize a set of transformed features into a feature cross token sequence, thereafter see LLM as a generator to learn patterns from the text they process, generate feature transformation tokens, and avoid searching in a large combination space. To address unsupervised transformation, we found that LLM agents exhibit a feature space diagnosis ability over tokenized data. We use such diagnosis ability to derive supervision knowledge from unsupervised data to guide the feature transformation process. We propose to unify generator-critic agents, in-context learning, and duet-play teaming to create "pseudo model", "pseudo objective", and "pseudo optimization" with only unlabeled data. In particular, the pseudo model is the generator agent that generates feature transformations given a dataset. The pseudo objective is the critic agent that diagnoses a dataset to generate feature space improvement advices as "textual gradient", which is equivalent to deriving optimization direction (i.e., gradient descent) from unsupervised data. The pseudo optimization is duet-play teaming between the critic agent and the generator agent, in which the advices of the critic agent are utilized to augment the in-context learning prompt of the generator agent in order to transform better feature space. The two agents team together and iteratively duet-play the same process.

Summary of proposed approach. We propose a duet-play generator-critic agent teaming framework to derive supervision from unsupervised data for fast unsupervised feature transformation. The framework includes three steps: 1) *The critic step:* the critic agent diagnoses semantic relationships and data distribution properties to generate advice for improving feature spaces; 2) *The generation step:* the generator agent tokenizes features, operators, and transformations and leverages in-context learning to produce a tokenized transformed feature set based on critic agent-augmented prompts; 3) *Iterative refinement:* a feedback loop between the critic agent and generator agents ensures continuous improvement of the generated features through semantic and structural alignment. In addition, this framework can be generalized from critic-augmented generation to human-agent collaborative generation, by replacing the critic agent with human experts. Finally, extensive experiments demonstrate our method is extremely efficient while accurate, highlighting its practical potential for EUFT.

2 Problem Definition

We utilize LLMs as agents for unsupervised feature generation. This approach enables the automated generation of meaningful features and improves performance in downstream tasks. Formally, given a dataset $\mathcal{D} = \{X, y\}$ and an operation set $\mathcal{O}$, where X is the original feature set and y is the corresponding label. Here, y is only used for the testing process. Our framework employs a two-stage approach for feature generation using LLMs. First, an critic LLM analyzes the original data set and then provides generation feedback to guide how to enhance the feature space, represented as $\theta = k(X)$, where k is the function notation of the critic. Second, we employ a generator LLM to perform feature generation. The generating feature space is represented by $\hat{\mathcal{X}} \sim g(\cdot|X, \mathcal{O}, \theta)$, which incorporates the original features and the critic insights. This two-stage process is repeated iteratively, with each step refining the feature space to enhance its representation for downstream tasks.

3 Generator-critic Feature Transformation

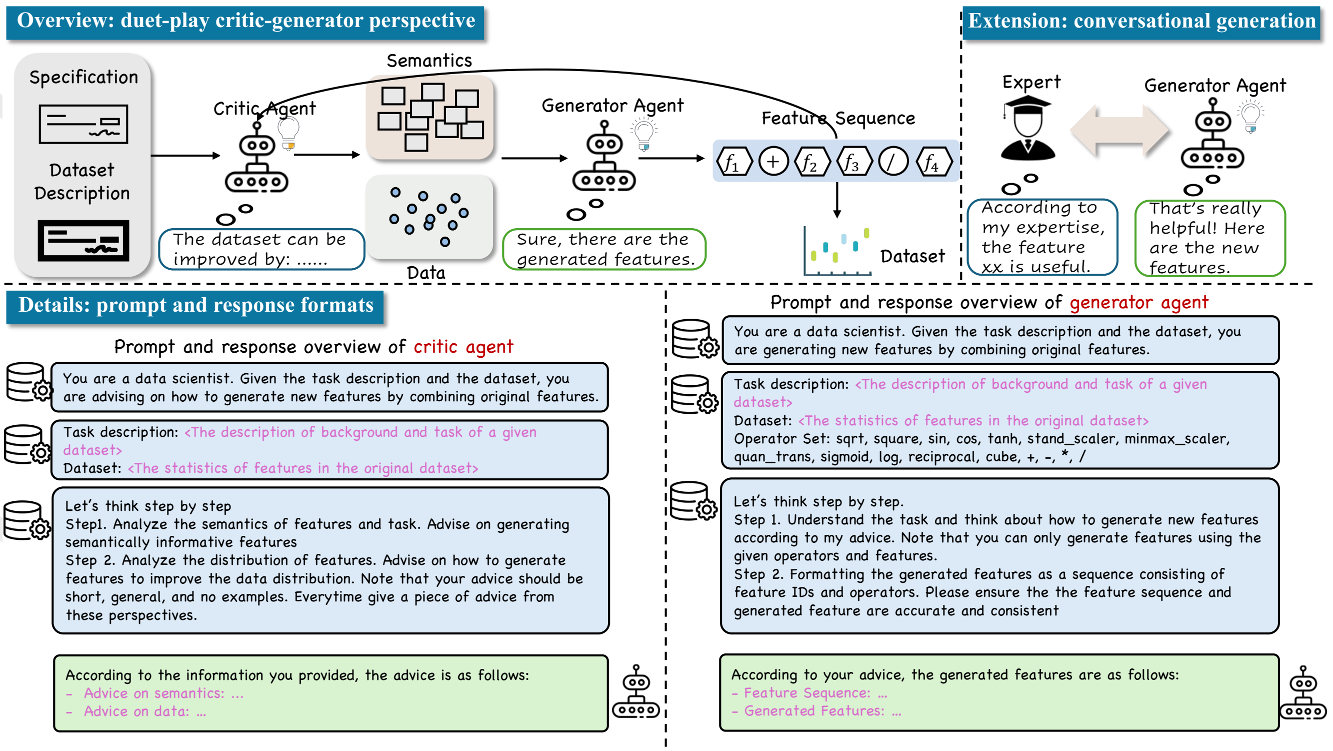
3.1 Framework Overview

Figure 1 shows our generator-critic LLM agents framework for unsupervised duet-play feature transformation includes: 1) the critic step: develop diagnosis and advice on generating meaningful features; 2) the generation step: feedback-driven feature generation; 3) the iterative refinement step.

In **Step 1**, given a data set, the critic agent aims to diagnose the dataset from both semantic and structural perspectives. Our idea is to leverage the general knowledge and reasoning abilities of LLMs to uncover insights into feature interactions, data relationships, and potential strategies for data augmentation. Specifically, the critic agent performs two key analyses: (i) *Semantic Analysis*: examining feature descriptions and task objectives to derive meaningful interactions and transformations. (ii) *Structural Analysis*: assessing the distribution and completeness of features to reason how to transform a feature space that aligns with the downstream task. Step 1 is to output a textual description of how to transform a dataset to make it AI ready. The benefit of this step is that the semantic, structure, and distribution-aware feature space transformation advices can inspire the generator agent into precise directions for generating informative and relevant features.

In **Step 2**, the generator agent regards a feature as a token, an operator as a token, and a transformed feature as a token segment (e.g., $f_1 * f_2$). Building upon this symbolic representation, the generator agent tokenizes a set of transformed features as a token sequence $(f1 * f2), log(f3), (f4/f5)$. The task of feature transformation is reformulated into generating a feature transformation token sequence given a dataset, which is achieved by the generator agent. This formulation enables straightforward reconstruction of a transformed dataset from the original dataset. The benefits of designing the generator agent are (i) efficiency: in-context prompting ensures rapid generation without sacrificing quality; (ii) adaptability: the generator agent can dynamically adapt to diverse datasets and leverage both prior knowledge and task-specific feedback from the critic agent; (iii) traceability: the idea of using symbolic token sequences to represent actionable feature transformations and using GenAI to learn and generate can facilitate the transparency and reproducibility of feature transformations.

In **Step 3**, to improve the robustness of the generated features, our framework iterates a feedback loop between the critic LLM agent and generator LLM agent. By comparing generated features with semantic rules and structural patterns identified by the critic LLM agent, we iteratively refine the


Overview: duet-play critic-generator perspective
Specification
Dataset Description
Critic Agent
Semantics
Data
Generator Agent
Feature Sequence
f_1 + f_2 f_3 / f_4
Dataset
The dataset can be improved by: ......
Sure, there are the generated features.
Extension: conversational generation
Expert
Generator Agent
According to my expertise, the feature xx is useful.
That's really helpful! Here are the new features.
Details: prompt and response formats

Prompt and response overview of critic agent

You are a data scientist. Given the task description and the dataset, you are advising on how to generate new features by combining original features.

Task description: <The description of background and task of a given dataset>
Dataset: <The statistics of features in the original dataset>

Let's think step by step
Step1. Analyze the semantics of features and task. Advise on generating semantically informative features
Step 2. Analyze the distribution of features. Advise on how to generate features to improve the data distribution. Note that your advice should be short, general, and no examples. Everytime give a piece of advice from these perspectives.

According to the information you provided, the advice is as follows:
- Advice on semantics: ...
- Advice on data: ...

Prompt and response overview of generator agent

You are a data scientist. Given the task description and the dataset, you are generating new features by combining original features.

Task description: <The description of background and task of a given dataset>
Dataset: <The statistics of features in the original dataset>
Operator Set: sqrt, square, sin, cos, tanh, stand_scaler, minmax_scaler, quan_trans, sigmoid, log, reciprocal, cube, +, –, *, /

Let's think step by step.
Step 1. Understand the task and think about how to generate new features according to my advice. Note that you can only generate features using the given operators and features.
Step 2. Formatting the generated features as a sequence consisting of feature IDs and operators. Please ensure the the feature sequence and generated feature are accurate and consistent

According to your advice, the generated features are as follows:
- Feature Sequence: ...
- Generated Features: ...

Figure 1: Framework overview. We implement feature generation through a duet-play generator-critic framework. We also extend it to a conversational generation manner.

feature space by coordinating the critic agent and the generator agent to converge toward ensuring semantic coherence, structural integrity, predictive utility, and format compatibility with downstream tasks, all in an unsupervised setting.

3.2 The Critic Step

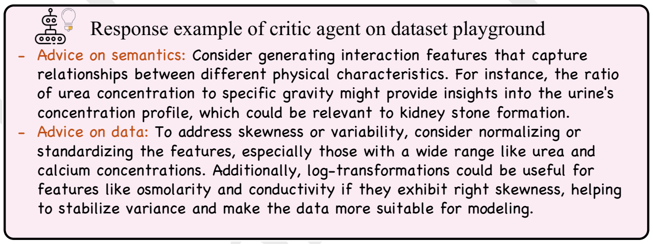
Why the critic agent matters? Firstly, a feature space is complex because features can vary in dimensionality (e.g., high dimensionality), correlation (e.g., nonlinear, interdependent, or context-specific), type (e.g., categorical, numerical, ordinal, ratio), and information properties (e.g., scale, redundancy, noise, or overfitting). This complexity makes it challenging to capture meaningful pattern, thus, it highly necessitates an automated tool to optimize meaningful feature representations. Secondly, traditional feature transformation relies on supervised feedback (e.g., task-specific predictive accuracy or feature importances) to guide optimization. In unsupervised settings, there are no explicit labels, making supervised feedback and optimization direction unavailable. The critic agent bridges this gap by evaluating datasets comprehensively, offering interpretable advice from both semantic and data distribution perspectives.

Leveraging AGI diagnosis of feature space as "textual gradient" of in-context feature transformation optimization. One strategy is to use a single LLM agent for generative feature transformation. However, the single agent needs to implicitly achieve both reasoning (i.e., diagnosing issues of a feature space and identifying improvement directions of feature space) and generation (i.e., generating token sequences of symbolic feature transformation actions). This strategy introduces uncertainty in coupling reasoning and generation toward optimal and requires the agent to precisely self-identify optimization direction. Our idea is to leverage a critic LLM agent to generate issue diagnosis and improvement advice of feature space as textual gradients, in order to provide unique optimization direction contexts within the prompt to adjust the generation agent's behavior.

Step 1: Semantic Diagnosis of Feature Space. The critic agent, trained on extensive textual corpora, can infer semantic relationships between input variables and target outputs. In particular, given the true names of predictors (X) and the response (Y), we prompt the critic agent to capture the semantic connections and contexts between X and Y to inspire the generator agent to create effective and interpretable features.

Step 2: Distribution Diagnosis of Feature Space. Besides, we prompt the critic agent to evaluate the underlying data distributions of feature space, in order to perceive whether the classification patterns or decision boundaries of data are discriminative and easy to learn. We exploit the perceived distribution information to inspire the critic agent to think how features can be transformed to reshape data distributions so classification patterns are well separated.

Integrating Semantic and Distributional Diagnosis. By integrating both the semantic diagnosis and the distributional diagnosis, the critic agent delivers well-rounded and actionable contexts to augment the prompt of the generator agent for feature space improvement. In this way, we can ensure that feature generation aligns with both the structural properties and semantic relationships of the dataset. Figure 2 presents a response example of the critic agent.



Response example of critic agent on dataset playground

- Advice on semantics: Consider generating interaction features that capture relationships between different physical characteristics. For instance, the ratio of urea concentration to specific gravity might provide insights into the urine's concentration profile, which could be relevant to kidney stone formation.
- Advice on data: To address skewness or variability, consider normalizing or standardizing the features, especially those with a wide range like urea and calcium concentrations. Additionally, log-transformations could be useful for features like osmolarity and conductivity if they exhibit right skewness, helping to stabilize variance and make the data more suitable for modeling.

Figure 2: We provide a response example of critic agent on the dataset playground. We obtain advice from semantic and data perspectives.

3.3 The Generation Step

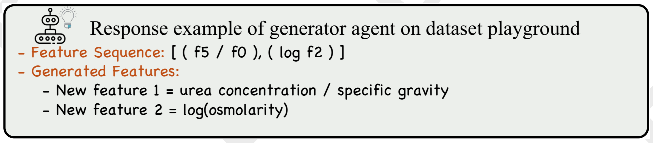
Why a generator agent matters? In prior literature, feature transformation achieved by manual transformations, supervised transformations (e.g., reinforcement, evolutionary), unsupervised transformations (e.g., PCA). However, manual methods are not generalizable and incomplete, as they heavily rely on domain and empirical experiences. Supervised methods require labeled data and need to search a vast space. Unsupervised methods are based on a strong assumption of straight linear feature correlation. The success of LLM shows it is appealing to model language knowledge as token sequences and reformulate predictive tasks as generative AI to regress next token. Following a similar spirit, we propose to represent mechanism-unknown feature space knowledge into symbolic sequential tokens. For instance, a transformed feature set ($\frac{f_1}{f_2}$, $f_1 - f_2$, $\frac{f_1+f_2}{f_1}$) is seen as a feature operator cross token sequence "$(f_1/f_2), (f_1 - f_2), ((f_1 + f_2)/f_1)$EOS". In other words, feature transformation can be viewed as a token generation task. Moreover, LLM exhibits in-context and few-shot learning abilities, thus, we can teach LLM to learn feature knowledge by demonstrating a list of feature transformation sequence examples in an instructional prompt.

Integrating feature-operator cross tokenization and in-context learning for fast and unsupervised feature transformation. Our idea is to see features and operators as tokens , and a transformed feature as a token segmentation of feature-operator crosses. We regard feature transformation as a generative AI task. LLM is specialized in sequential token generation. Its in-context learning ability allows us to incorporate critic LLM to learn complex feature space knowledge and the optimization direction of sequence generation, by demonstrating relevant background information, specific instructions, examples of desired outputs, clear task definition, and structured formatting. Such reformulation can help to achieve efficient and unsupervised feature transformation,

Step 1: Tokenize Feature Transformations. The generator agent tokenizes each transformation into a sequence of tokens (e.g., $f_1 * f_2/f_3$). This symbolic representation not only enables the LLM token generation of feature transformation, but also facilitates the transparent tracking of transformations.

Step 2: In-context Prompting for Rapid Generation. The generator agent leverage the outputs from the critic agent to construct in-context prompts to dynamically generate high-quality features. By combining the task-specific guidance from the critic agent with the generic knowledge knowledge of the generator agent, the generator agent generates feature transformations without supervised labels. Figure 3 demonstrates an example of the generator agent response.

There are two major benefits of using the generator agent: 1) **Efficiency:** in-context learning ensures rapid feature generation without compromising on quality. 2) **Traceability:** The use of symbolic token sequences enhances transparency and reproducibility, making the feature transformations easily interpretable and verifiable.



Response example of generator agent on dataset playground

- Feature Sequence: [(f5 / f0), (log f2)]
- Generated Features:
  - New feature 1 = urea concentration / specific gravity
  - New feature 2 = log(osmolarity)

Figure 3: We present a response example of generator agent on the dataset playground. The feature sequence represent a dataset and the generated features interpret the semantic meanings.

3.4 The Iterative Refinement Step

Leveraging critic agent-augmented generation for iterative improvement. In each iteration, the critic agent generates semantic and distributional diagnosis of feature space, along with feature transformation advices, as enriched contexts. We then leverage the critic agent-generated diagnosis and advices to augment the in-context learning prompt of the generator agent. During such iterative refinement, the generator agent dynamically adapts to diverse datasets by integrating task-specific diagnosis and advices from the critic agent and AGI knowledge.

3.5 From Critic-Augmented Generation to Human-Agent Collaborative Generation

The emergence of Reinforcement Learning from Human Feedback (RLHF) demonstrates the significance of human feedback for diverse and insightful generation [Yu *et al.*, 2024; Wang *et al.*, 2023b]. Traditionally, feature transformation is effective with domain knowledge and empirical experience under the guidance of human experts.

We want to highlight that our critic-generator framework can be converted into human-agent collaborative generation. Specifically, we can replace the critic agent with a user or a human expert to implement customized feature transformation. The human expert can write domain-specific instructions to help LLM incorporate human thinking and expert knowledge into the in-context learning. For instance, human expert can provide a domain-specific and personalized instruction to the LLM, by inputting an instruction like: "Feature f_3 is interesting. Please generate new variants of f_3."

4 Experimental Results

4.1 Experimental Setup

Data Descriptions. We utilize 12 public datasets that contain task descriptions and feature names from Kaggle and OpenML to conduct experiments.

Evluation Metrics. We employ Random Forest as the downstream ML model. The accuracy score of the predictions is used to evaluate the performance of methods.

Dataset	Source	Original	TTG	AutoFeat	GRFG	OpenFE	CAAFE	LPFG
balance	OpenML	0.750	0.781	0.688	0.813	0.781	0.781	**0.906**
cmc	OpenML	0.507	0.518	0.504	0.542	0.534	0.507	**0.561**
credit-g	OpenML	0.780	0.78	0.772	0.788	0.764	0.772	**0.792**
diabetes	OpenML	0.786	0.797	0.786	0.797	0.797	0.786	**0.813**
tic-tac-toe	OpenML	0.667	0.708	0.625	0.750	0.625	0.792	**0.833**
pc1	OpenML	0.935	**0.939**	0.935	**0.939**	0.935	**0.939**	**0.939**
airlines	OpenML	0.638	0.640	0.628	0.644	0.614	0.624	**0.650**
jungle	OpenML	0.848	0.852	0.856	0.856	0.850	0.846	**0.860**
health	Kaggle	0.742	0.740	0.740	0.742	0.748	0.736	**0.752**
pharyngitis	Kaggle	0.680	0.711	0.680	**0.719**	0.695	0.711	**0.719**
spaceship	Kaggle	0.744	0.754	0.746	0.748	**0.756**	0.754	0.750
playground	Kaggle	0.750	0.750	0.721	0.750	0.712	0.712	**0.760**

Table 1: Overall comparison of different models on across 12 datasets. We bold the best results and underline the second-best results.

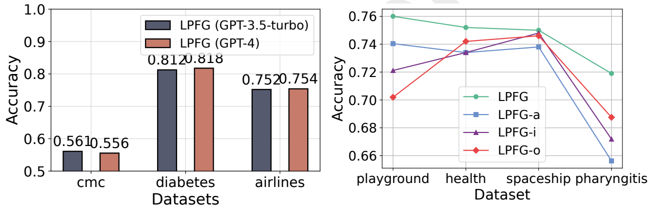
Baselines and variants. To demonstrate the effectiveness of our method, We compare LPFG with 5 widely-used models in feature transformation: 1) **TTG** [Khurana *et al.*, 2018] formulates feature transformation as a graph and searches via reinforcement learning; 2) **AutoFeat** [Horn *et al.*, 2020] expands the feature space and performs feature selection to retain the meaningful features; 3) **GRFG** [Wang *et al.*, 2022] builds a multi-agent framework to automatically generate new features and optimize; 4) **OpenFE** [Zhang *et al.*, 2023] proposes a feature boosting method and a two-stage pruning algorithm to implement expand-reduce feature engineering; 5) **CAAFE** [Hollmann *et al.*, 2024] leverages the in-context learning ability of LLM to conduct feature transformation.

To comprehensively evaluate the necessity of each component of LPFG, we introduce variant models: 1) **LPFG-a** adopt the supervised performance of a given feature set on the downstream ML model as the feedback; 2) **LPFG-i** leverage the importance obtained from Random Forest to guide feature transformation; 3) **LPFG-o** remove the critic agent and assign the generator to handle both reasoning and generation.

4.2 Experimental Results

Overall Performance. This experiment aims to answer: *Is the proposed method effective for improving downstream ML model performance?* We compare the proposed method with the baselines on 12 datasets. Note that the LLM-based methods (i.e., CAAFE and LPFG) use GPT-3.5-turbo through API. Table 1 shows that LPFG, as an unsupervised method, outperforms all supervised baselines on most datasets. The underlying driver is that the critic agent can comprehensively evaluate the datasets and give useful advice for feature transformation. Besides, an intriguing observation from the dataset *playground* reveals that LPFG is capable of identifying effective optimization directions even on challenging datasets, whereas the baseline methods fail to improve performance. On the one hand, that is probably because the generator-critic framework is more noise-resistant and robust. On the other hand, a potential distribution shift may affect the performance of supervised methods. However, LPFG demonstrates the capability to achieve more reliable optimization by conducting a comprehensive evaluation of the dataset.

The Impact of Different LLMs This experiment aims to answer: *Does the choice of LLM model affect the performance of our method?* We adopt different LLMs to build the system and analyze the influence of the LLM backbone on performance. Figure 4(a) presents the results of LPFG with *GPT-3.5-turbo* and *GPT-4o* respectively. The results indicate that the performance of both models remains comparable. Despite substituting with a more advanced LLM, the accuracy exhibits only marginal variation. The reason is two-fold: For the critic agent, evaluating the dataset from semantic and distributional perspectives is a relatively straightforward task, as the general knowledge embedded in the LLM often enables it to provide constructive suggestions. For the generator, its role is merely to follow the critic agent's guidance and explore potential feature combinations. Consequently, high-level intelligence is not required, making the implementation of LPFG more cost-effective and practical.



(a) LLM Impact

(b) Guidance Impact

Figure 4: Ablation study. (a) We study the impact of using different LLMs in LPFG. (b) We investigate the performance of generator guided by different information.

The Impact of Guidance for Generation This experiment aims to answer: *Is the advice from critic agent better than supervised signals (e.g., accuracy and feature importance) for guiding generation?* To validate the effectiveness of the critic agent in facilitating feature transformation, we introduce variant models LPFG-a and LPFG-i, which utilize downstream model accuracy and feature importance as guidance, respectively, while LPFG-o operates without any feedback. As illustrated in Figure 4(b), the first key observation is the noticeable performance decline of LPFG-o compared to

LPFG. A underlying driver is that a single LLM struggles to balance reasoning and generation, making it difficult to generate meaningful features. Furthermore, models guided by accuracy and feature importance also experience performance degradation, even underperforming compared to models without any guidance on several datasets. This can be attributed to the generator agent's inability to effectively interpret non-traceable information within a few-iteration feature transformation process, thereby restricting its capacity to guide the generation of new features. In contrast, the actionable advice generated by the critic agent effectively aids in optimizing the feature space without requiring extensive time for feedback interpretation and potential space exploration.

Robustness Check This experiment aims to answer: *Is our method robust when collaborate with different downstream models?* We employ various downstream ML models, i.e., XGBoost (XGB), Support Vector Machine (SVM), K-Nearest Neighborhood (KNN), Decision Tree (DT), AdaBoost (ADA), to study the robustness of the proposed method on the dataset *diabetes*. Figure 5 illustrates that LPFG achieves the best performance, except when the downstream task employs XGB. This can be explained by LPFG is task-agnostic, as it operates in an unsupervised manner. In contrast, supervised methods are affected by their performance on specific models, leading to greater fluctuations. As an unsupervised plug-in model, LPFG's robustness highlights its practical value in real-world machine learning tasks.

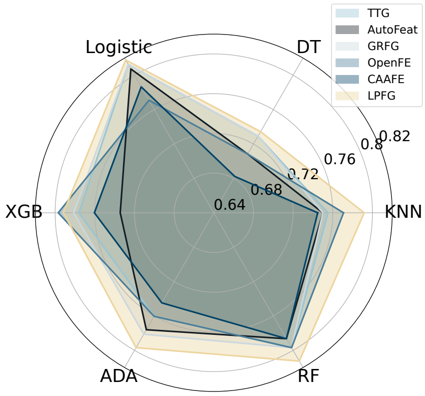


Figure 5: Robustness check. On *diabetes*, we investigate the robustness of the proposed method when different downstream ML models are employed.

Time Complexity Study *Is our model efficient in feature transformation?* We further investigate the time complexity of the proposed method on 6 datasets. Since GRFG and TTG are reinforcement learning-based methods that require time for search within the solution space, we primarily focus on comparing the time complexity of lightweight models. Figure 6 presents the feature transformation time for different models. It can be observed that LPFG demonstrates consistently short and stable generation time. In contrast, the time cost of AutoFeat and OpenFE significantly fluctuates with the size of the dataset. As another LLM-based model, CAAFE requires slightly more time and exhibits minor fluctuations. This can be explained by the fact that LPFG avoids repeatedly computing downstream model accuracy, and the inference time of the LLM through API is fast and consistent.

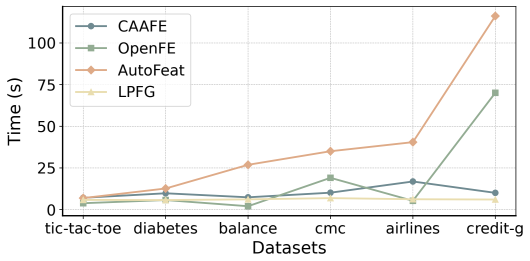


Figure 6: Time complexity study. We compare the costing time (s) for feature tran on 6 datasets.

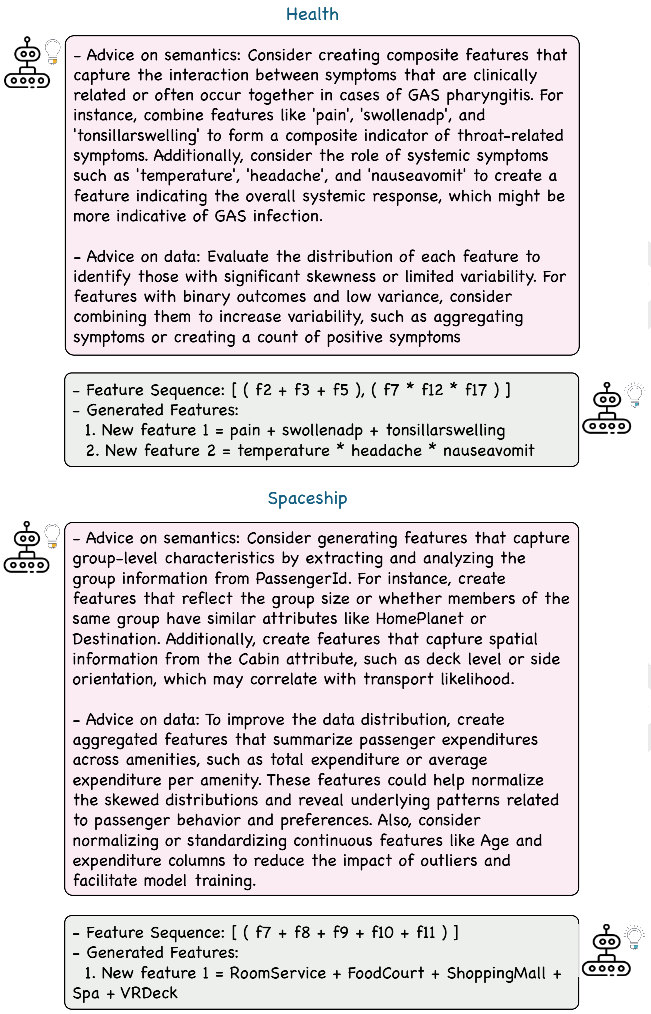
Case Study: Duet-Play Generator-Critic Framework for Feature Transformation This experiment aims to answer: *How can the teaming LLMs collaborate on feature transformation?* We illustrate the communication process between the critic agent and the generator agent in Figure 7. The critic agent provides actionable suggestions for optimizing the feature space, while the generator agent considers these suggestions to make the final decision. Through these examples, we observe two key points: consistency and interpretability. On one hand, the generated features align with certain parts of the advice, indicating that the critic agent provides meaningful guidance in feature transformation. On the other hand, our approach not only enables a concise feature set representation through feature sequences with minimal tokens but also allows for the interpretation of the generated features, enhancing transparency and understanding.

Case Study: Conversational Feature Transformation This experiment aims to answer: *How can users achieve customized feature transformation in a conversational manner?* We present several cases of conversational feature transformation in Figure 8 to explore the effectiveness and flexibility of our framework. Instead of adopting an automatic critic agent, we proactively input feature transformation requirements or suggestions, allowing the generator to operate in a more customized manner. This interactive and adaptive approach offers a novel solution for feature transformation, enhancing the practicality and engagement of LPFG.

5 Related work

5.1 Feature Transformation

Feature transformation is an essential task in Data-Centric AI [Ying *et al.*, 2025; Gong *et al.*, 2025a; Gong *et al.*, 2025b; Ying *et al.*, 2024]. It aims to generate new features to improve the feature space and enhance downstream models. For example, ExploreKit [Katz *et al.*, 2016] creates an extensive set of candidate features by integrating information from the original features, and Cognito [Kanter and Veeramachaneni, 2015] investigates diverse feature construction options using a hierarchical and selective approach. Furthermore, various strategies for discrete decision-making are applied in this task. EAAFE [Zhu *et al.*, 2022] proposes to leverage an evolutionary algorithm to improve feature transformation. GRFG [Wang *et al.*, 2022] designs three reinforcement learning agents to collaborate in generating new features. Recently, generative feature transformation has been proposed



Health

- Advice on semantics: Consider creating composite features that capture the interaction between symptoms that are clinically related or often occur together in cases of GAS pharyngitis. For instance, combine features like 'pain', 'swollenadp', and 'tonsillarswelling' to form a composite indicator of throat-related symptoms. Additionally, consider the role of systemic symptoms such as 'temperature', 'headache', and 'nauseavomit' to create a feature indicating the overall systemic response, which might be more indicative of GAS infection.

- Advice on data: Evaluate the distribution of each feature to identify those with significant skewness or limited variability. For features with binary outcomes and low variance, consider combining them to increase variability, such as aggregating symptoms or creating a count of positive symptoms

- Feature Sequence: [(f2 + f3 + f5), (f7 * f12 * f17)]
- Generated Features:
  1. New feature 1 = pain + swollenadp + tonsillarswelling
  2. New feature 2 = temperature * headache * nauseavomit

Spaceship

- Advice on semantics: Consider generating features that capture group-level characteristics by extracting and analyzing the group information from PassengerId. For instance, create features that reflect the group size or whether members of the same group have similar attributes like HomePlanet or Destination. Additionally, create features that capture spatial information from the Cabin attribute, such as deck level or side orientation, which may correlate with transport likelihood.

- Advice on data: To improve the data distribution, create aggregated features that summarize passenger expenditures across amenities, such as total expenditure or average expenditure per amenity. These features could help normalize the skewed distributions and reveal underlying patterns related to passenger behavior and preferences. Also, consider normalizing or standardizing continuous features like Age and expenditure columns to reduce the impact of outliers and facilitate model training.

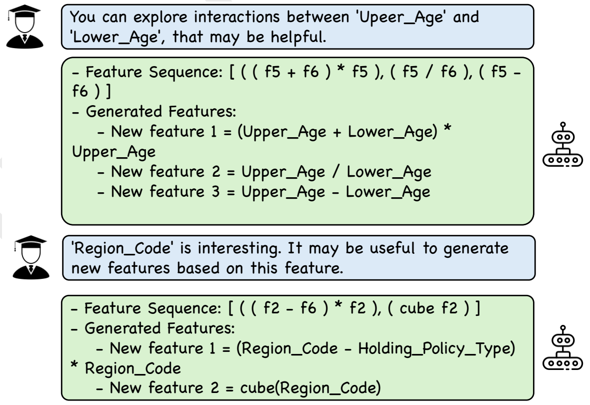
- Feature Sequence: [(f7 + f8 + f9 + f10 + f11)]
- Generated Features:
  1. New feature 1 = RoomService + FoodCourt + ShoppingMall + Spa + VRDeck

Figure 7: Case study. We provide some examples of generator-critic generation process. The critic agent suggests optimization paths, and the generator agent selects and takes actions on the most promising one.

and achieves promising performance. MOAT [Wang *et al.*, 2023a] formulates feature transformation as a sequential generation task. They embed the feature sets into a continuous space and perform the gradient-steered search for the optimal feature set.

5.2 Task-specific LLM

With LLM demonstrating competitive performance across a wide range of fields, there is an increasing number of works focusing on the application of LLM for specific tasks [Wang *et al.*, 2025]. Aug-iModels [Singh *et al.*, 2023] enhances linear models with LLM embeddings and decision trees with LLM-generated features, improving performance and interpretability in NLP tasks and showing promise in neuroscience. The paper [Li *et al.*, 2023] reviews LLM applications in finance, proposes a decision framework for selecting solutions based on data, compute, and performance, and discusses key limitations to guide responsible financial AI use. LLM is also utilized to mine and understand relationships in graph data and is applied to recommendation tasks [Wang *et al.*, 2024]. There are also methods utilizing LLM for data science. For instance, CAAFE [Hollmann *et al.*, 2024] leverages an LLM to iteratively generate additional semantically meaningful features for tabular datasets using the dataset's description. ELLM-FT [Gong *et al.*, 2024] proposes to integrate evolutionary algorithm and LLM to generate new feature sets by few-shot prompting. This paper differs from the previous LLM-based methods in two key aspects: 1) it is under an unsupervised setting which is very challenging for feature transformation; 2) we leverage two specialized LLM agents for dataset diagnosis and feature generation, rather than a single LLM. 3) we extend the method to conversational feature transformation, providing a novel interactive way for this task.



You can explore interactions between 'Upeer_Age' and 'Lower_Age', that may be helpful.

- Feature Sequence: [((f5 + f6) * f5), (f5 / f6), (f5 - f6)]
- Generated Features:
  - New feature 1 = (Upper_Age + Lower_Age) * Upper_Age
  - New feature 2 = Upper_Age / Lower_Age
  - New feature 3 = Upper_Age - Lower_Age

'Region_Code' is interesting. It may be useful to generate new features based on this feature.

- Feature Sequence: [((f2 - f6) * f2), (cube f2)]
- Generated Features:
  - New feature 1 = (Region_Code - Holding_Policy_Type) * Region_Code
  - New feature 2 = cube(Region_Code)

Figure 8: Case study. We provide some examples of conversational feature transformation. LPFG can create new features according to personal requirements.

6 Conclusion

We introduce a duet-play generator-critic LLM agents model. Our approach implements unsupervised feature generation in three steps: 1) we employ a critic agent for dataset diagnosis. Leveraging the general knowledge, it provides feature set improvement suggestions from both semantic and distributional perspectives in an unsupervised manner; 2) we build a generator agent to create new features. Based on the advice from the critic agent, the generator makes the final feature set optimization decision and generates new features in a sequential formulation; 3) We iterate the feedback loop between the critic agent and generator agent, continuously refining the feature space. The proposed framework achieves unsupervised dataset diagnosis and improvement. By teaming two specialized LLM agents, we avoid repeated feature combination space exploration and implement robust and efficient feature set optimization in few iterations. Our method is also extended to a novel conversational feature generation formulation. Replacing the critic agent with a human expert, we integrate the expertise into LLM and build a flexible and interactive system for feature generation. Finally, extensive experiments demonstrate the effectiveness, robustness, efficiency, and traceability of our method.

Acknowledgments

This paper is supported by the National Science Foundation (NSF) via the grant numbers: 2426340, 2416727, 2421864, 2421865, 2421803, and National Academy of Engineering Grainger Foundation Frontiers of Engineering Grants.

References

[Gong *et al.*, 2024] Nanxu Gong, Chandan K Reddy, Wangyang Ying, and Yanjie Fu. Evolutionary large language model for automated feature transformation. *arXiv preprint arXiv:2405.16203*, 2024.

[Gong *et al.*, 2025a] Nanxu Gong, Sixun Dong, Haoyue Bai, Xinyuan Wang, Wangyang Ying, and Yanjie Fu. Agentic feature augmentation: Unifying selection and generation with teaming, planning, and memories. *arXiv preprint arXiv:2505.15076*, 2025.

[Gong *et al.*, 2025b] Nanxu Gong, Zijun Li, Sixun Dong, Haoyue Bai, Wangyang Ying, Xinyuan Wang, and Yanjie Fu. Sculpting features from noise: Reward-guided hierarchical diffusion for task-optimal feature transformation. *arXiv preprint arXiv:2505.15152*, 2025.

[Hollmann *et al.*, 2024] Noah Hollmann, Samuel Müller, and Frank Hutter. Large language models for automated data science: Introducing caafe for context-aware automated feature engineering. *Advances in Neural Information Processing Systems*, 36, 2024.

[Horn *et al.*, 2020] Franziska Horn, Robert Pack, and Michael Rieger. The autofeat python library for automated feature engineering and selection. In *Machine Learning and Knowledge Discovery in Databases: International Workshops of ECML PKDD 2019, Würzburg, Germany, September 16–20, 2019, Proceedings, Part I*, pages 111–120. Springer, 2020.

[Kanter and Veeramachaneni, 2015] James Max Kanter and Kalyan Veeramachaneni. Deep feature synthesis: Towards automating data science endeavors. In *2015 IEEE international conference on data science and advanced analytics (DSAA)*, pages 1–10. IEEE, 2015.

[Katz *et al.*, 2016] Gilad Katz, Eui Chul Richard Shin, and Dawn Song. Explorekit: Automatic feature generation and selection. In *2016 IEEE 16th international conference on data mining (ICDM)*, pages 979–984. IEEE, 2016.

[Khurana *et al.*, 2018] Udayan Khurana, Horst Samulowitz, and Deepak Turaga. Feature engineering for predictive modeling using reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[Li *et al.*, 2023] Yinheng Li, Shaofei Wang, Han Ding, and Hang Chen. Large language models in finance: A survey. In *Proceedings of the fourth ACM international conference on AI in finance*, pages 374–382, 2023.

[Singh *et al.*, 2023] Chandan Singh, Armin Askari, Rich Caruana, and Jianfeng Gao. Augmenting interpretable models with large language models during training. *Nature Communications*, 14(1):7913, 2023.

[Wang *et al.*, 2022] Dongjie Wang, Yanjie Fu, Kunpeng Liu, Xiaolin Li, and Yan Solihin. Group-wise reinforcement feature generation for optimal and explainable representation space reconstruction. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 1826–1834, 2022.

[Wang *et al.*, 2023a] Dongjie Wang, Meng Xiao, Min Wu, Yuanchun Zhou, Yanjie Fu, et al. Reinforcement-enhanced autoregressive feature transformation: Gradient-steered search in continuous space for postfix expressions. *Advances in Neural Information Processing Systems*, 36:43563–43578, 2023.

[Wang *et al.*, 2023b] Yuanhao Wang, Qinghua Liu, and Chi Jin. Is rlhf more difficult than standard rl? a theoretical perspective. *Advances in Neural Information Processing Systems*, 36:76006–76032, 2023.

[Wang *et al.*, 2024] Xinyuan Wang, Liang Wu, Liangjie Hong, Hao Liu, and Yanjie Fu. Llm-enhanced user-item interactions: Leveraging edge information for optimized recommendations. *arXiv preprint arXiv:2402.09617*, 2024.

[Wang *et al.*, 2025] Xinyuan Wang, Yanchi Liu, Wei Cheng, Xujiang Zhao, Zhengzhang Chen, Wenchao Yu, Yanjie Fu, and Haifeng Chen. Mixllm: Dynamic routing in mixed large language models. *arXiv preprint arXiv:2502.18482*, 2025.

[Ying *et al.*, 2024] Wangyang Ying, Haoyue Bai, Kunpeng Liu, and Yanjie Fu. Topology-aware reinforcement feature space reconstruction for graph data. *arXiv preprint arXiv:2411.05742*, 2024.

[Ying *et al.*, 2025] Wangyang Ying, Cong Wei, Nanxu Gong, Xinyuan Wang, Haoyue Bai, Arun Vignesh Malarkkan, Sixun Dong, Dongjie Wang, Denghui Zhang, and Yanjie Fu. A survey on data-centric ai: Tabular learning from reinforcement learning and generative ai perspective. *arXiv preprint arXiv:2502.08828*, 2025.

[Yu *et al.*, 2024] Tianyu Yu, Yuan Yao, Haoye Zhang, Taiwen He, Yifeng Han, Ganqu Cui, Jinyi Hu, Zhiyuan Liu, Hai-Tao Zheng, Maosong Sun, et al. Rlhf-v: Towards trustworthy mllms via behavior alignment from fine-grained correctional human feedback. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13807–13816, 2024.

[Zhang *et al.*, 2023] Tianping Zhang, Zheyu Aqa Zhang, Zhiyuan Fan, Haoyan Luo, Fengyuan Liu, Qian Liu, Wei Cao, and Li Jian. Openfe: automated feature generation with expert-level performance. In *International Conference on Machine Learning*, pages 41880–41901. PMLR, 2023.

[Zhang *et al.*, 2024] Xinhao Zhang, Jinghan Zhang, Banafsheh Rekabdar, Yuanchun Zhou, Pengfei Wang, and Kunpeng Liu. Dynamic and adaptive feature generation with llm. *arXiv preprint arXiv:2406.03505*, 2024.

[Zhu *et al.*, 2022] Guanghui Zhu, Shen Jiang, Xu Guo, Chunfeng Yuan, and Yihua Huang. Evolutionary automated feature engineering. In *Pacific Rim International Conference on Artificial Intelligence*, pages 574–586. Springer, 2022.