Review

Not peer-reviewed version

Dynamic Feature Engineering Through Reinforcement and Prompt Based Learning

Tanmay Karthik [*]

Posted Date: 28 May 2025

doi: 10.20944/preprints202505.2193.v1

Keywords: Machine Learning; Reinforcement learning

Article

Dynamic Feature Engineering Through Reinforcement and Prompt Based Learning

Tanmay Karthik

Affiliation 1; tanmaytadala@gmail.com

Abstract: Feature engineering is an important part of machine learning processes that has a big impact on how well models work, how easy they are to understand, and how effective they are overall. Feature selection and transformation are often done with filter, wrapper, and embedding techniques, but they often require manual heuristics and subject knowledge. They are also ineffective in contexts characterized by high dimensionality and complexity. Recent studies have explored automated techniques utilizing big language models and reinforcement learning to address these limitations. This paper presents a thorough and critically analyzed review of cutting-edge research on reinforcement learning-based feature selection, reinforcement learning-driven feature production, and LLM-guided feature optimization. Three primary paradigms of technique are recognized. Initially, feature selection is conceptualized as a collaborative or directed decision-making challenge employing interactive and multi-agent reinforcement learning methodologies. These strategies assign agents to features and optimize long-term rewards based on domain-specific importance, redundancy, or model precision. Combinatorial Multi-Armed Bandits (CMAB) represent a computationally efficient alternative that facilitates scalable and effective feature selection with minimal learning overhead, being a component of the second paradigm [1]. In the third type, LLMs are employed to either derive effective reward functions or generate novel features. They accomplish this through the utilization of reasoning-based prompts, external knowledge repositories, and prototype alignment. This work also addresses unresolved issues in bias management, computational overhead, and generalization to unfamiliar domains, as well as underexplored gaps, including the necessity for hybrid frameworks that integrate the exploration efficiency of reinforcement learning with the semantic reasoning of large language models.

Keywords: machine learning; reinforcement learning

---

1. Introduction

It is often more important to choose the right characteristics for the input feature space or come up with new ones in machine learning than to choose the specific model design or training method. Traditional methods like filter, wrapper, and embedded approaches have had a lot of success, but they have some problems, such as not being able to be scaled up, not being easy to understand, and relying on human heuristics [2].The demand for automated, adaptable, and explainable feature engineering techniques has increased markedly as datasets become more high-dimensional, sparse, and multimodal.

Recent research have addressed this challenge by defining feature optimization as an independent learning problem. Reinforcement learning (RL) techniques employ task-specific reward functions to model features as decision-making agents that progressively optimize selection strategies. Certain methods proficiently investigate and utilize subsets of features through lightweight variations such as multi-armed bandits (MAB) [1]. While these methods offer adaptability and flexibility, they may struggle to generalize beyond the numerical realm or with semantic comprehension.

Large language models (LLMs) have demonstrated growing significance in simultaneous feature generation and evaluation through natural language reasoning and domain expertise. LLM-based systems can propose pertinent, elucidated characteristics derived from text metadata, external corpora, or human preferences utilizing tools such as Tree-of-Thought prompting, Retrieval-Augmented Generation (RAG), and few-shot reward modeling [3,4].

This study provides a comprehensive assessment of diverse works across multiple paradigms, concentrating on strategies that automate feature selection or creation using reinforcement learning and large language models. We categorize them by architectural and methodological themes rather than analyzing each individually, evaluate their design trade-offs, and analyze their strengths in practical applications. This research highlights significant deficiencies, including justice, cost-effectiveness, and insufficient hybridization.

2. Previous Work

Historically, feature engineering has been a manual process reliant on statistical heuristics, domain expertise, and iterative error-based methodologies. Initially, filter, wrapper, and embedding approaches were employed to automate this task; however, they exhibited limited flexibility in rapidly changing situations or those with numerous dimensions. Recent research has redefined feature selection as a learning problem by employing reinforcement learning (RL) and information-theoretic frameworks to successively pick subsets based on reward signals associated with model accuracy and redundancy. Despite frequently functioning as a black box with restricted post-hoc interpretability, methodologies such as INVASE [5] and L2X [6] have progressed this domain by implementing instance-specific feature selection through actor-critic mechanisms and mutual information approximations.

A lot of studies have shown that RL-based automation can do more than just picking. It can also do generation. Khurana et al [7] came up with one of the first RL-based AutoFE models, but their system needed transformation templates that had to be made by hand and couldn't understand semantics. Recent projects like AutoML-Zero [8] try to build feature creation logic from scratch. However, they are very hard to code and don't have the reasoning structure that is needed for results that can be understood. The goal of neural-symbolic systems like AutoFeature [9] is to close this gap by combining neural operators with structured thinking.

Utilizing existing domain knowledge in natural language, large language models (LLMs) offer a semantic abstraction layer for this area. Conversely, numerous LLM-based generators rely solely on task-level correctness as the metric of success. This study asserts that this type of input is inadequate; hence, feature traceability and redundancy should also be guiding factors for generation. Hybrid systems in which LLMs generate proposals and RL agents validate them are becoming increasingly essential for achieving semantic depth and optimization rigor.

3. Methodologies

The current increase in automated feature engineering research indicates a shift from static, human heuristics to models capable of reasoning, adapting, and learning transformations or selections. In the analyzed studies, we discern three primary methodological prototypes that characterize feature optimization: (i) reinforcement learning for feature selection, (ii) reinforcement learning for feature production, and (iii) LLM-guided feature optimization. These categories indicate not only varying structures but also fundamentally divergent perspectives on the representation of decision processes within the feature space.

The initial group frames the application of reinforcement learning (RL) for feature selection as a sequential decision-making problem. The objective is to train agents to identify which attributes should be retained or eliminated to enhance downstream model performance. Each feature is conceptualized as an autonomous agent inside the multi-agent reinforcement learning framework introduced by Liu et al [2], receiving feedback through task accuracy, redundancy penalties, and other reward signals. Gao et al [10] expand upon this by incorporating external guidance from KBest filters and decision trees. Distinguishing "hesitant" from "assertive" agents enhances interpretability and convergence by offering the latter targeted direction in initial training phases. Li et al [1] address the issue using combinatorial multi-armed bandits (CMAB), thereby circumventing the intricacies of comprehensive reinforcement learning with a more streamlined methodology. This formulation enables rapid and scalable feature selection by incorporating inherent methods to balance mutual information-based relevance and redundancy.

The second collection of works develops new features through mathematical modifications rather than picking existing ones. These methodologies are predicated on the conviction that latent structures within the data can be elucidated through novel representations generated by operations such as addition, logarithmic transformation, or interaction. Three sequential RL agents select two feature groups and an operation to apply between them, so creating GRFG, a group-wise feature generator. Mathematical similarity and mutual information govern these actions. InHRecon, introduced by Zhang et al. [11], establishes a hierarchical agent framework that modularizes transformation, feature targeting, and operation selection. Their technique ensures that the generated interactions accurately represent genuine second-order effects through the utilization of H-statistics, hence enhancing interpretability.

Large language models (LLMs) are used instead of numerical models for semantic reasoning and text-informed feature engineering in the third practical class. Still, LLMs are useful in more than one way in these systems. In Large language model based Feature Generation [12] and Text Informed Feature Generation [3],the LLM mostly functions as a feature generator suggesting changes depending on stimuli, metadata, or acquired knowledge. Proto-RM [4] learns preference-aligned scoring systems from minimal input using LLMs as part of the reward modeling pipeline instead. These methodological clusters provide a more pertinent framework for comparison than individual paper summaries. Each category exhibits trade-offs in generalization, interpretability, computational expense, and relevance to practical issues. Despite their sensitivity to reward shaping, RL-based selectors are flexible and lightweight. Despite their computational expense, generative agents effectively document interactions. Despite incorporating domain reasoning, LLM-guided models encounter challenges related to repeatability and bias mitigation.

3.1. Reinforcement Learning Strategies for Dynamic Feature Selection

Reinforcement learning-based automated feature selection is a rational advancement from static filter or wrapper methods to dynamic, feedback-oriented selection. The primary idea is to depict the selection of a feature subset as a sequential decision-making process, wherein the learner observes a state (the selected feature subset) and receives delayed incentives (such as subsequent accuracy) upon completion. This domain generates three distinct methodologies, each addressing the challenges of reward sparsity, interpretability, and scalability.

Mutual information (MI) is frequently utilized in the assessment of redundancy and relevance. Specifically, redundancy penalizes overlapping characteristics within the selected subset, whereas relevance promotes traits that align with the target label.:

$$\text{Redundancy} = \frac{1}{|S|^2} \sum_{i,j \in S} I(f_i; f_j) \tag{1}$$

$$\text{Relevance} = \frac{1}{|S|} \sum_{i \in S} I(f_i; y) \tag{2}$$

These metrics help guide the learning process toward informative and diverse subsets, balancing model performance and interpretability.

First representative work models in multi-agent reinforcement learning each as an independent agent. Every agent learns a policy by a common environment evaluating the combined subset on a downstream action and generates binary judgments either select or drop. Especially in early peroid, this structure promotes distributed exploration but suffers from delayed and sparse rewards even if it is naturally parallelizable. The authors aid to reduce feature redundancy by means of group-level normalisation and correlation aware reward components.

Upper Confidence Bound (UCB): The CMAB approach uses UCB to balance exploration and exploitation. The reward estimate for feature arm i at time t is computed as [1]:

$$\text{Score}_i = \mu_i + \sqrt{\frac{2 \log t}{n_i}} \tag{3}$$

Here, μ_i is the mean reward of arm i, n_i is the number of times it has been selected, and t is the total time steps. This formulation encourages selection of promising but under-explored features.

A second way builds on the limitations of pure MARL by interacting with standard models to add monitoring from outside the RL loop. This design labels agents as "assertive" or "hesitant" based on how much they trust each other. Decision trees and statistical filters are common ways to train people who aren't sure what to do. This makes convergence faster, selection quality better, and learning more stable, especially when there isn't much data. The system can switch between supervised guidance and autonomous policy learning on the fly, combining the flexibility of current RL with the readability of classical logic.

In contrast to the agent-intensive formulations previously mentioned, a third approach conceptualizes each characteristic as an independent arm, hence streamlining the decision framework through combinatorial multi-armed bandits (CMAB). Unlike comprehensive reinforcement learning approaches, these techniques do not represent environmental states or long-term policy trajectories. CMAB optimizes feature subset selection by utilizing short-term reward estimates, commonly through generative Beta sampling or confidence upper bounds. While CMAB models are significantly quicker to train, more interpretable, and more resilient in low-resource or latency-sensitive contexts, their absence of state transitions renders CMAB less expressive than deep RL in sequential decision-making scenarios. They also streamline tuning and acknowledge the effort in reward design. CMAB remains a viable option for tabular datasets when feature independence is presumed or tolerated, despite its inability to model inter-feature interdependence over time steps. The disparities in method design between MARL and bandit-based techniques are mostly influenced by this distinction in expressiveness and efficiency. These models significantly vary in agent architecture, reward granularity, and computing demands. Table 1 presents a comparative summary.

Table 1. Comparative Summary of Reinforcement Learning Models for Feature Selection.

Method	Agent Architecture	Reward Signal	Strengths	Limitations
MARL	One agent assigned per feature	Global task accuracy combined with redundancy penalties	Enables coordinated decision-making among feature-specific agents; highly parallelizable	Suffers from reward sparsity and delayed learning feedback
Interactive RL	Agents with external trainer supervision	Hybrid signal incorporating model-driven heuristics and task-specific accuracy	Achieves rapid convergence and improved interpretability via guided learning	Relies on external model feedback, reducing generalizability
CMAB	Each feature modeled as an arm in a bandit setting	Mutual information and accuracy-based evaluation metrics	Low computational overhead; strong baseline performance under tight constraints	Limited expressiveness; incapable of modeling higher-order feature interactions

3.2. *Structured Feature Construction via Reinforcement Learning*

Feature generation methods produce new attributes by mathematical or statistical transformations, while feature selection techniques concentrate on determining an appropriate subset of existing features. Reinforcement learning (RL) offers a methodical approach to autonomously guide agents through processes such as addition, multiplication, or logarithmic transformations. These methods preserve interpretability while enhancing model performance by linking each generated feature to its original components.

Group-wise Reinforced Feature Generation (GRFG) [11] presents an interesting method generating three consecutive reinforcement learning agents. The first two agents choose groupings of original features while the third agent chooses a transformation operation (e.g., addition, logarithmic transformation, or cross-product). Since GRFG combines two feature groups instead of aggregating two independent features, therefore producing numerous new features in a single phase, the group-wise interaction of GRFG is unique. This arrangement guarantees better quality of reward feedback and accelerates training. Cosine similarity and mutual knowledge help the agents to guarantee both relevance and diversity.

InHRecon employs a hierarchical reinforcement learning architecture to segment feature creation into three sequential decisions: selecting the transformation operation, identifying the first feature, and then determining the second feature. The approach assesses whether features are numerical or categorical and employs H-statistics to determine the strength of second-order interactions. This ensures the correct processes are executed. These design choices are most effective in contexts where specialized knowledge is limited and complex relationships are difficult to identify manually. InHRecon evaluates both the validity of transformations and the quality of interactions to identify the optimal balance between interpretability and generalization capability.

Feature Interaction Strength: To measure the second-order interaction between features, InHRecon uses Friedman's H-statistic [12]:

$$H^2 = \frac{\mathbb{V}[\mathbb{E}[Y \mid X_1, X_2] - \mathbb{E}[Y \mid X_1] - \mathbb{E}[Y \mid X_2]]}{\mathbb{V}[Y]} \tag{4}$$

This measures how much of the label variation is due to interaction effects. This makes sure that the features that are created show real relationships instead of just noise.

GRFG and InHRecon aim to enhance the performance of subsequent models by developing characteristics that are structured and comprehensible. However, due to their distinct methods of organizing agents, disseminating guidance messages, and providing feedback, they exhibit varying trade-offs, as illustrated in Table 2.

Table 2. Comparison of RL-Based Feature Generation Methods.

Method	Agent Design	Feature Construction Strategy	Guidance Signal Used	Strengths	Limitations
GRFG	Cascaded multi-agent pipeline (Group1 $\rightarrow$ Operation $\rightarrow$ Group2)	Constructs cross-feature interactions by applying selected operations to grouped subsets	Mutual information metrics and cosine similarity scores guide generation	Enables efficient batch creation of cross features; yields stronger task-aligned rewards	Dependent on effective clustering; performance sensitive to group coherence
InHRecon	Hierarchical decision chain (Operation $\rightarrow$ Feature1 $\rightarrow$ Feature2)	Sequentially constructs binary feature interactions by selecting operator and operands in staged decisions	H-statistics to quantify interaction strength; enhanced with data type constraints	Provides interpretable, fine-grained control over generated interactions	Introduces complexity via multi-step reasoning; requires tighter agent synchronization

In GRFG, the term **cascaded agents** refers to a sequential reinforcement learning setup where each agent makes a decision that feeds into the next stage of the feature generation pipeline.

3.3. Language Guided Feature Engineering with External Knowledge

Large language models (LLMs) are a completely different way of thinking about semantic reasoning, while reinforcement learning methods explore feature space through reward-optimized exploration. Instead of just using numbers or task-specific rewards, LLM-guided systems use language, metadata, and outside knowledge to suggest, rate, or rank new features. These models are great at giving tabular data meaning and encoding logic at the expert level. Three new studies show that LLMs can do three different things: preference-based reward modeling (Proto-RM), knowledge-informed augmentation (TIFG), and feature building (LFG).

The LFG paradigm sees each change in a feature as a task that needs to be thought out. Tree-of-Thought (ToT) prompting is used by some LLM agents to come up with candidate features, explain their reasoning, and change outputs based on input they get later. Include a Monte Carlo Tree Search (MCTS) system to help choose the best choice paths based on model performance metrics, like F1 score or accuracy. This method mixes symbolic reasoning with probabilistic search so that LLMs can look into a number of transformations—for example, log(income), weight × age without coming up with ideas that are too similar or don't make sense. LFG works well with many models, like KNN, MLP, and Random Forest, with little labeled guidance.

The idea that real-world traits can have secret meanings in their names or dataset metadata is built upon by TIFG [3]. Retrieval-Augmented Generation (RAG) looks for task-related ideas on Wikipedia or other sources, then asks the LLM to put together things like "density = population / land area" or "BMI = weight / height2." This method works best for financial and healthcare datasets where subject knowledge is not built into the raw numbers. TIFG supports thinking and justifying in more than one way, which creates features that are both understandable and new.

Proto-RM [4] shifts the focus from making features to evaluating and aligning those features. It teaches a prototypical reward model by using input from a small number of human comparisons (chosen from features or outputs that were rejected) to make prototype vectors. Then, new examples are given marks based on how much they look like the prototypes that were bought. This model makes reward accuracy better in low-label settings, especially for LLM output alignment, preference modeling, or safety tuning in chat systems.

Prototype-Guided Loss Function: Proto-RM optimizes a composite loss to align model outputs with human preference [4]:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{reward}} + \rho_d \cdot \mathcal{L}_{\text{diversity}} \tag{5}$$

$\mathcal{L}_{\text{reward}}$ in this formula promotes the model to provide preferred outputs like human preference rankings or accurate classifications—higher scores, which are usually based on pairwise comparisons. By encouraging variance among learnt prototype representations, the $\mathcal{L}_{\text{diversity}}$ term helps avoid mode collapse and enhances generalization to a variety of input sources. The trade-off between alignment accuracy and representation spread is managed by the hyperparameter ρ_d.

Table 3. Comparative Analysis of LLM-Guided Feature Generation and Evaluation Frameworks.

Method	Core Functionality	Reasoning Strategy	Strengths	Limitations
LFG	Semantic feature construction through iterative reasoning	Tree-of-Thought prompting combined with Monte Carlo Tree Search	Enables structured transformations with minimal labeled supervision; supports multi-step reasoning chains	Highly sensitive to prompt phrasing and search depth; computationally intensive
TIFG	Context-aware feature synthesis using external corpora	Retrieval-Augmented Generation (RAG)	Produces domain-grounded features aligned with task context; leverages large-scale external knowledge	Heavily dependent on retrieval quality; results may lack reproducibility across datasets
Proto-RM	Reward-guided evaluation of LLM outputs via human preference alignment	Prototype-based Reward Modeling using contrastive supervision	Promotes efficient, interpretable learning of human-aligned reward signals; applicable for fine-tuning LLM behavior	Requires curated preference comparisons; limited direct use for generative feature construction

4. Applications

The implementation of the diversity in design between RL-based and LLM-guided feature engineering techniques directly leads to distinct strengths in practical domains. Each method, ranging from policy and preference modeling to healthcare and finance, is tailored to meet the specific operational constraints, data structures, and interpretability requirements.

Particularly appropriate for healthcare data pipelines are joint feature-instance selection models such the one detailed in [3]. Apart from redundant features, these databases often include useless or chaotic data including mislabeled or outlier patient records. Introduced in DAIRS, the dual-agent reinforcement learning method helps to selectively filter both axis characteristics and cases, thereby enhancing clinical robustness and generalization. Furthermore, the ability to track choices fits legal standards for interpretability in medical uses.

Transformer-based architectures for feature weighting, such as those employed in high-cardinality tabular domains, are inherently adaptable to financial risk modeling and fraud detection tasks. These parameters necessitate the dynamic reweighting of behavior metrics, transaction patterns, and credit history over time. While the transformer paper was excluded from our study, the feature relevance modeling utilizing structured agents, as examined in GRFG [11], offers comparable value for these financial tabular jobs.

Policy and insurance-related utilization cases frequently benefit from text-enhanced, semantically grounded features. This is demonstrated by TIFG [3], which generates features that are informed by external corpora such as Wikipedia or internal documentation. Consequently, applications such as risk profiling or policy scoring are enabled from both numerical attributes and textual descriptions. These methods are particularly beneficial when domain knowledge is concealed in unstructured forms.

Finally, methods such as Proto-RM [4] facilitate preference-aligned tasks, such as reinforcement learning from human feedback (RLHF) and chatbot customization. In sparse-feedback or low-label systems, it is imperative to acquire robust reward models from a limited number of samples. The prototypical-based architecture of Proto-RM is particularly relevant in dialog agents, educational platforms, and interactive ranking systems, where feedback is frequently implicit or overly-noisy.

5. Challenges

Current techniques have a lot of problems that make them less useful in real life, even though automated feature optimization has a lot of promise. One persistent problem is the lack of standardized samples designed to test feature generation and selection methods. In most cases, techniques are tested on UCI tabular datasets or domain-specific collections, which don't provide a reliable basis for judging the quality of the features that are created. Without widely accepted benchmarks, it is hard to compare algorithms fairly across fields and study groups.

Another significant distinction is revealed by combining language model-based thinking with reinforcement learning. Semantic generalization and knowledge retrieval from LLMs are rarely combined with structured exploration and optimization from RL techniques. Contemporary models typically select one theory while disregarding the advantages of the other. Hybrid models, where LLMs propose or defend changes while RL agents validate and refine those decisions over time, may prove useful for future studies.

Bias and fairness are also largely ignored. The generated features may incorporate implicit societal biases from external corpora or training distributions, especially in text-informed or feedback-guided environments such as TIFG [3] or Proto-RM [4]. This raises serious concerns in high-stakes industries like hiring, lending, and healthcare. Adding fairness constraints or adversarial debiassing to the reward model could provide excellent guidance.

In the end, computational cost remains a significant obstacle, especially for models that rely on cascaded agents or multiple rounds of LLM reasoning, like GRFG [11] and LFG [12]. These approaches might not be feasible in low-resource environments or for time-sensitive applications. They can be made more deployable and with less overhead by using surrogate modeling, curriculum learning, or policy distillation. All of these issues point to the need for more ethical, efficient, and modular frameworks that can be applied to different industries while maintaining interpretability and computational viability.

6. Future Work

Recent advances in automated feature engineering suggest that systems that combine the structured optimization of reinforcement learning with the generative reasoning of large language models will become more integrated and semantically aware. Despite advances in feature transformation and selection, RL-based methods usually lack semantic context. In contrast, domain aware reasoning can be achieved by LLM guided systems like TIFG and LFG [3,12], but they require costly feedback cycles. Future research might look into hybrid architectures in which RL agents use reward guided learning based on domain knowledge to validate features suggested by LLMs. Prototype based alignment has potential in low-label settings, but it needs to be extended to multimodal tasks [4]. Addressing bias propagation, energy cost, and generalization across datasets are still crucial. The next generation of approaches must eventually bring together statistical learning, ethical constraints, and symbolic reasoning for interpretable and efficient feature space construction.

7. Conclusions

This study examined current developments in automated feature space building via the lens of big language models and reinforcement learning. The study arranged methods into three methodological archetypes they are RL-based selection, RL-guided generation, and LLM-driven semantic reasoning instead of seeing present efforts as isolated. Class affects the trade-offs among interpretability, scalability, and generalization. Whereas generative RL agents expose higher-order interactions, bandit-based selectors and multi-agent systems offer modular control over redundancy and relevance. LLM-guided methods improve semantic alignment even if they usually rely only on task-level feedback and have a considerable computational overhead.

Many of these approaches are still domain fragile, sensitive to hyperparameter tuning, and computationally expensive in practice even if their predictive performance and adaptation obviously improve. Very few systems nowadays combine knowledge-based feature building with symbolic reward modeling. Furthermore understudied are evaluation consistency, traceability, and bias. Automating feature design across several disciplines calls for a more complete approach using statistical decision processes and semantic abstraction. Future feature engineering will be influenced by hybrid systems combining the rigor of RL with the intuition of LLMs, so blending symbolic exploration with semantic reasoning.

References

1. Liu, K.; Huang, H.; Zhang, W.; Hariri, A.; Fu, Y.; Hua, K. Multi-armed bandit based feature selection. *Proceedings of the 2021 SIAM International Conference on Data Mining (SDM)* **2021**.
2. Chen, J.; Song, L.; Wainwright, M.; Jordan, M. Learning to explain: An information-theoretic perspective on model interpretation. *International conference on machine learning* **2018**.
3. Zhang, X.; Liu, K. Tifg: Text-informed feature generation with large language models. *2024 IEEE International Conference on Big Data (BigData)* **2024**.
4. Zhang, J.; Wang, X.; Jin, Y.; Chen, C.; Zhang, X.; Liu, K. Prototypical reward network for data-efficient rlhf. *arXiv preprint arXiv:2406.06606* **2024**.
5. Real, E.; Liang, C.; So, D.; Le, Q. Automl-zero: Evolving machine learning algorithms from scratch. *International conference on machine learning* **2020**.
6. Pulicharla, M.R. Neurosymbolic AI: Bridging neural networks and symbolic reasoning **2025**.
7. Fan, W.; Liu, K.; Liu, H.; Ge, Y.; Xiong, H.; Fu, Y. Interactive reinforcement learning for feature selection with decision tree in the loop. *IEEE Transactions on Knowledge and Data Engineering* **2021**.
8. Yoon, J.; Jordon, J.; Van der Schaar, M. INVASE: Instance-wise variable selection using neural networks. *International conference on learning representations* **2018**.
9. Wang, D.; Fu, Y.; Liu, K.; Li, X.; Solihin, Y. Group-wise reinforcement feature generation for optimal and explainable representation space reconstruction. *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining* **2022**.
10. Khurana, U.; Samulowitz, H.; Turaga, D. Feature engineering for predictive modeling using reinforcement learning. *Proceedings of the AAAI Conference on Artificial Intelligence* **2018**.
11. Zhang, X.; Zhang, J.; Rekabdar, B.; Zhou, Y.; Wang, P.; Liu, K. Dynamic and adaptive feature generation with llm. *arXiv preprint arXiv:2406.03505* **2024**.
12. Liu, K.; Fu, Y.; Wu, L.; Li, X.; Aggarwal, C.; Xiong, H. Automated feature selection: A reinforcement learning perspective. *IEEE Transactions on Knowledge and Data Engineering* **2021**.