

MoFA: LLM-Driven Reasoning for Constraint-Aware Feature Selection in Industrial ML Systems

Yuhang Zhou
Meta Platforms, Inc.
USA

Spilios Evmorfos
Meta Platforms, Inc.
USA

Qiao Liu
Meta Platforms, Inc.
USA

Weiwei Li
Meta Platforms, Inc.
USA

Gedi Zhou
Meta Platforms, Inc.
USA

Zhuokai Zhao
Meta Platforms, Inc.
USA

Gökalp Demirci
Meta Platforms, Inc.
USA

Qifei Wang
Meta Platforms, Inc.
USA

Tingting Wang
Meta Platforms, Inc.
USA

Abhishek Kumar
Meta Platforms, Inc.
USA

Ke Li
Meta Platforms, Inc.
USA

Mingyi Wang
Meta Platforms, Inc.
USA

Serena Li
Meta Platforms, Inc.
USA

Mingze Gao
Meta Platforms, Inc.
USA

Xiangjun Fan
Meta Platforms, Inc.
USA

Lizhu Zhang
Meta Platforms, Inc.
USA

Jiayi Liu
Meta Platforms, Inc.
USA

Abstract

Feature selection is a crucial step in large-scale industrial machine learning systems, directly affecting model accuracy, efficiency, and maintainability. Traditional feature selection methods rely on labeled data and statistical heuristics, making them difficult to apply in production environments where labeled data are limited and multiple operational constraints must be satisfied. To address this, we propose **Model Feature Agent** (MoFA), a large language model-driven framework that performs sequential, reasoning-based feature selection using both semantic and quantitative feature information. MoFA incorporates feature definitions, importance scores, correlations, and metadata (e.g., feature groups or types) into structured prompts and selects features through interpretable, constraint-aware reasoning. We evaluate MoFA in three real-world industrial applications: (1) True Interest and Time-Worthiness Prediction, where it improves accuracy while reducing feature group complexity; (2) Value Model Enhancement, where it discovers high-order interaction terms that yield substantial engagement gains in online experiments; and (3) Notification Behavior

Prediction, where it selects compact, high-value feature subsets that improve both model accuracy and inference efficiency. Together, these results demonstrate the practicality and effectiveness of LLM-based reasoning for feature selection in real production systems.

1 Introduction

Feature selection is a fundamental component of industrial machine learning systems, directly influencing model accuracy, interpretability, and efficiency [2, 16, 21]. In large-scale production environments such as recommendation and notification systems, models often rely on hundreds of heterogeneous features collected from user behavior, content attributes, and system signals [9]. Selecting the right subset of features is therefore essential for both predictive performance and maintainability. However, traditional feature selection approaches depend heavily on labeled training data and numerical statistics [18, 28], making them difficult to apply when labeled data are scarce or when additional business or operational requirements must be satisfied.

Recent advances in large language models (LLMs) provide new opportunities for feature reasoning. LLMs are capable of understanding both the semantic meaning and quantitative context of features described in natural language [5, 14, 23]. This capability opens a new paradigm for feature selection that extends beyond purely statistical criteria to incorporate contextual and domain-specific knowledge. However, existing LLM-based approaches focus solely on optimizing predictive utility while overlooking practical constraints, such as capacity limits or maintainability, and often

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
Conference'17, Washington, DC, USA

© 2026 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 978-x-xxxx-xxxx-x/YYYY/MM
<https://doi.org/10.1145/nnnnnnnn.nnnnnnn>

fail to leverage the rich metadata naturally available in production systems. These limitations motivate a more general and constraint-aware framework for LLM-driven feature selection in real-world industrial settings.

In this work, we propose Model Feature Agent (MoFA), an LLM-based framework for feature selection in industrial systems. MoFA formulates feature selection as a sequential reasoning process: at each step, the LLM receives structured information about the task objective, model configuration, current and available features, and auxiliary requirements, and then selects the next most suitable feature along with interpretable reasoning. To address the scalability challenges in industrial settings with thousands of features, we introduce a Divide-and-Conquer strategy that partitions the feature space into manageable buckets for parallel selection followed by a global refinement phase.

We evaluate MoFA through three real-world industrial use cases. In the True Interest and Time-Worthiness Prediction task, MoFA improves accuracy while reducing feature group complexity. In the Value Model Enhancement task, it identifies meaningful higher-order interaction terms, and online experiments show that incorporating these terms into the value model yields substantial gains in key engagement metrics. In the Notification Behavior Prediction task, MoFA selects compact, high-value feature subsets from a large heterogeneous feature space, improving both model accuracy and inference efficiency. Together, these results demonstrate that LLM-based reasoning can serve as a practical and effective tool for feature selection in modern production systems, bridging the gap between interpretability, flexibility, and real-world scalability.

2 Related Work

Feature Selection with Classical Models. Feature selection aims to identify the most informative subset of features that improves model generalization, interpretability, and efficiency [1, 7, 12, 27]. Existing methods are commonly categorized into three groups: **filter**, **wrapper**, and **embedded** approaches. Filter methods evaluate feature relevance independently of any specific model, typically using statistical measures such as correlation, mutual information, or variance scores [4, 6, 8, 17, 19]. Wrapper methods, in contrast, rely on iterative model training and evaluation to assess feature subsets [3, 15, 29, 30], which often yields higher accuracy but incurs substantial computational cost, such as sequential selection and recursive feature elimination [11, 12, 24]. Embedded methods integrate feature selection directly into the model training process, for example through sparsity-inducing regularization or tree-based importance measures [10, 28, 32]. Despite their effectiveness in data-rich scenarios, these traditional approaches require access to sufficient labeled data and cannot leverage the semantic or metadata information associated with each feature, limitations that motivate our use of large language models for more context-aware and flexible feature selection.

Feature Selection with LLMs. Recent studies have explored using LLMs to assist or automate feature selection through natural language reasoning and prompt-based evaluation. One line of work formulates feature selection as a classification task, where the LLM is prompted to determine whether a given feature is important for a specific prediction goal or to assign an importance score based

on textual descriptions [5, 14]. Another line of research treats it as a ranking problem, leveraging the LLM’s ability to re-rank features according to their relevance or contribution to model performance [14, 20, 22]. A third direction refines feature selection iteratively, incorporating feedback from downstream model results to improve subsequent selections [13, 31]. However, these approaches typically rely on labeled data and controlled settings, whereas our work targets real-world industrial scenarios with limited supervision, rich feature metadata, and multiple operational constraints.

3 MoFA Workflow

Feature selection plays a crucial role in industrial recommender systems, where models must balance multiple goals such as predictive accuracy, user engagement, and system efficiency. Traditional approaches mainly rely on numerical correlations or model-based importance scores, often ignoring the semantic and contextual meaning of features. As a result, these methods may fail to capture how different features relate to product objectives or operational constraints.

To address this gap, we propose the **MoFA** pipeline. As shown in Figure 1, MoFA leverages LLMs as reasoning agents that can analyze not only the descriptions of features but also their associated metadata, such as feature importance and correlations with key metrics. By integrating these heterogeneous sources of information, MoFA performs sequential, context-aware feature selection that aims to improve model performance while satisfying additional business requirements.

3.1 Problem Formulation

Let $\mathcal{F} = \{f_1, f_2, \dots, f_N\}$ denote the set of available features, and let $\mathcal{S}_t \subseteq \mathcal{F}$ denote the current feature subset selected at step t . Each feature f_i is represented as a tuple: $f_i = (\text{name}_i, \text{desc}_i, \text{meta}_i)$, where name_i and desc_i are the textual name and description of the feature, and meta_i contains auxiliary quantitative information (e.g., feature importance scores, correlation coefficients with target metrics, or group constraints).

Given a target objective function $\mathcal{O}$ (e.g., model performance, engagement lift, or a weighted combination of business metrics), the goal is to select a subset $\mathcal{S}^* \subseteq \mathcal{F}$ satisfying:

$$\mathcal{S}^* = \arg \max_{\mathcal{S} \subseteq \mathcal{F}} \mathcal{O}(\mathcal{S}) \quad \text{s.t.} \quad C(\mathcal{S}) = \text{true},$$

where $C(\mathcal{S})$ represents a set of auxiliary constraints such as capacity limits, feature group restrictions, or fairness requirements.

3.2 Scalable Sequential Selection via Divide-and-Conquer

In industrial settings where the total number of features N exceeds the context window of the LLM, we adopt a two-phase **Divide-and-Conquer** approach. Both phases utilize the same core sequential reasoning selection logic to maintain consistency and depth.

Phase 1: Bucketed Parallel Selection. We first partition the total feature set $\mathcal{F}$ into B disjoint buckets $\{\mathcal{B}_1, \dots, \mathcal{B}_B\}$. For each bucket $\mathcal{B}_b$, the MoFA agent performs sequential selection until it identifies a candidate subset $\mathcal{S}_b$ where $|\mathcal{S}_b| = K' > K/B$. This ensures that highly relevant features from each partition are preserved for global

evaluation while staying within the model’s context window during each reasoning step.

Phase 2: Global Synthesis and Refinement. The subsets are merged into a unified candidate pool $\mathcal{F}_{refine} = \bigcup_{b=1}^B \mathcal{S}_b$. The agent then performs the final round of sequential reasoning selection on $\mathcal{F}_{refine}$ to reach the target size K . By evaluating the top candidates from all buckets simultaneously, the LLM can reason about dependencies and redundancies across the entire filtered feature space, mitigating collinearity problems that might otherwise arise from independent bucket selections.

3.3 Sequential Selection via LLM Reasoning

The MoFA agent decomposes the selection process into a series of LLM-driven reasoning steps, where one feature is selected at each iteration. At step t , given the current feature set $\mathcal{S}_t$ and the remaining available features $\mathcal{A}_t$ (within a specific bucket or the refined pool), the LLM receives a structured prompt $\mathcal{P}_t$ that includes: (1) A description of the task and model objective; (2) The list of features currently in use, $\mathcal{S}_t$, along with their associated metadata; (3) The list of candidate features, $\mathcal{A}_t$, each annotated with semantic information; and (4) Optional auxiliary requirements $\mathcal{C}(\mathcal{S})$.

Formally, the LLM is prompted to output:

$$f_t^* = \arg \max_{f_i \in \mathcal{A}_t} \Phi_\theta(f_i | \mathcal{S}_t, \mathcal{O}, \mathcal{R}),$$

where Φ_θ denotes the LLM’s internal reasoning and decision function parameterized by θ . The selected feature f_t^* is then added to the current set:

$$\mathcal{S}_{t+1} = \mathcal{S}_t \cup \{f_t^*\}, \quad \mathcal{A}_{t+1} = \mathcal{A}_t \setminus \{f_t^*\}.$$

The process continues until the required number of features for that specific phase is reached.

Prompt Construction and Output Format. At each selection step, MoFA constructs a structured user prompt providing all necessary contextual information for informed decision-making. The LLM then reasons over these inputs to identify the next most suitable feature. Each reasoning step is decomposed into two parts: (1) the model’s natural language reasoning process, and (2) a structured selection output. The expected response format is:

Selected Feature: f_t^* , Reason: ρ_t

where ρ_t represents the explanation for why the selected feature f_t^* best satisfies the main objective $\mathcal{O}$ while adhering to the auxiliary constraints $\mathcal{C}$.

4 Experiments

We evaluate MoFA in three real-world industrial applications, covering different model architectures and feature domains, to assess its effectiveness and adaptability across practical settings.

4.1 Experimental Setup

Models and Baselines. For feature selection, we employ the Llama4-Maverick model as the backbone of the MoFA agent [25]. This model serves as the reasoning engine that sequentially evaluates and selects features based on the constructed prompts. To provide meaningful comparisons, we include two types of baselines. For

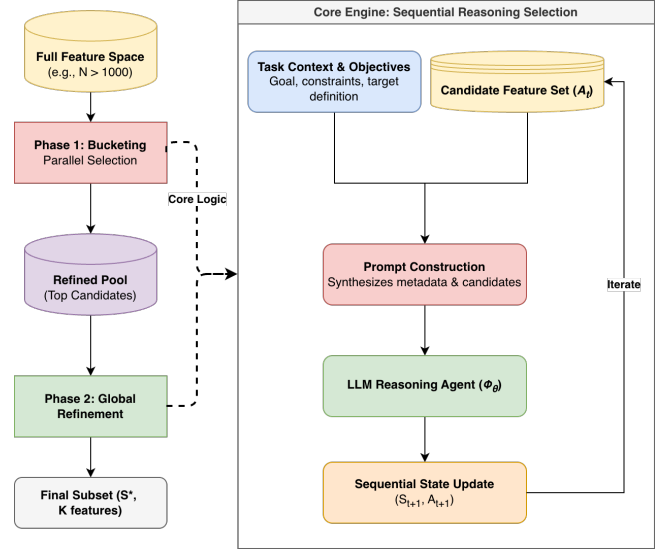


Figure 1: The MoFA pipeline with Divide-and-Conquer scaling. The process begins by partitioning the candidate pool into disjoint buckets (Phase 1). An LLM agent (Φ_θ) performing sequential reasoning first selects local candidate subsets from each bucket. These subsets are merged into a refined candidate pool, where the agent performs a final round of global sequential selection (Phase 2) to reach the target size K .

use cases with available offline training data, we adopt Lasso regression as a representative statistical feature selection method that leverages data-driven sparsity [28]. For use cases without offline data, common in real-world recommendation systems where only online signals are available, we construct a baseline by randomly selected features in online experiments. This setup allows us to assess the performance gain of MoFA under both data-rich and data-scarce environments.

4.2 Prediction of True Interest and Time-Worthiness

Background and Task. In this use case, we seek to predict whether a piece of content (post or short video) reflects a user’s *true interest* and whether it is *worth the user’s time*. To obtain supervision aligned with user sentiment, we randomly sample content exposures and trigger an in-product survey that asks users to judge (i) whether the item reflects their true interest and (ii) whether it was worth their time. Let $y^{\text{TI}} \in \{0, 1\}$ and $y^{\text{WT}} \in \{0, 1\}$ denote the corresponding binary labels. Each training instance i consists of a feature vector $x_i \in \mathbb{R}^N$ that concatenates user-level and content-level features, $x_i = [x_i^{(u)}; x_i^{(c)}]$, and the labels $(y_i^{\text{TI}}, y_i^{\text{WT}})$. The learned predictors are integrated into the recommendation stack as additional signals (e.g., ranking features) to improve downstream relevance.

Modeling and Feature-Group Requirement. We train logistic regression models for each label using a subset of features selected by MoFA. Concretely, for task $q \in \{\text{TI}, \text{WT}\}$ we estimate $p^{(q)}(x; \beta^{(q)}) =$

$\sigma((\beta^{(q)})^\top x_{S(q)})$, where $\sigma(\cdot)$ is the logistic link and $x_{S(q)}$ indexes the coordinates of x selected for task q . Features are partitioned into operational *groups* $\mathcal{G} = \{G_1, \dots, G_G\}$ (e.g., by owning team or data pipeline), and our auxiliary requirement is to *concentrate* the selected features into fewer groups to reduce cross-team dependencies and maintenance overhead.

Within MoFA, this auxiliary requirement is implemented by providing each candidate feature’s group identifier as part of its metadata (meta_{*i*}) in the prompt. During prompt construction, we explicitly instruct the LLM to prioritize selections that maintain fewer feature groups whenever possible. In this way, the LLM integrates the group information into its reasoning process, balancing predictive performance with operational simplicity. The resulting sequential, LLM-guided selection produces compact and maintainable feature sets that align well with practical engineering constraints.

Offline Experiment Setup. We evaluate MoFA on two prediction tasks derived from user survey data: *true interest* (TI) and *time-worthiness* (WT). The TI and WT dataset comprises 71,754 training instances and 17,939 test instances. Both tasks predict binary labels based on user responses to in-product surveys. The feature space consists of user-level and content-level features organized into operational groups maintained by different teams. We evaluate feature subsets of size $K \in \{20, 100, 200, 500\}$ from a total of 1,030 features to assess scalability across sparse and dense feature regimes. To manage this feature space within the LLM’s context window, we employ the Divide-and-Conquer strategy by partitioning the 1,030 features into $B = 5$ disjoint buckets. In Phase 1, the agent selects $K' = 1.5 \times (K/B)$ features from each bucket. In Phase 2, the agent performs a final sequential selection to reach the target K . For the Lasso baseline, we use L_1 -regularized logistic regression with regularization parameter $C = 0.1$, selecting the top- K features ranked by absolute coefficient magnitude.

Table 1 presents the performance and feature group utilization for both methods across different feature budget sizes for both prediction tasks.

Predictive Accuracy. The relationship between MoFA and Lasso varies by task and budget. For TI prediction, MoFA outperforms Lasso at smaller budgets (+0.63% at $K = 20$, +0.16% at $K = 100$), demonstrating that semantic reasoning effectively identifies informative features under tight constraints. At larger budgets ($K \geq 200$), both methods converge to similar performance (within 0.5% AUC). For WT prediction, Lasso achieves slightly higher test AUC across all budgets, though the gap is less than 0.5% in most cases. Despite modest AUC differences, MoFA delivers substantial operational benefits through feature group consolidation, as discussed below.

Feature Group Efficiency. The most significant advantage of MoFA is its ability to dramatically reduce the number of feature groups required, thereby lowering cross-team coordination overhead and maintenance complexity. For TI prediction, MoFA achieves a **30% reduction** in groups at $K = 200$ (61 vs 87) and a **54% reduction** at $K = 500$ (85 vs 187). For WT prediction, the gains are even more pronounced: a **10% reduction** at $K = 200$ (76 vs 84), and a dramatic **69% reduction** at $K = 500$ (96 vs 313 groups).

These results demonstrate that MoFA’s LLM-based reasoning successfully identifies features that balance predictive signal with

True Interest				
K	Method	Groups	Train AUC	Test AUC
20	Lasso	9	74.49	71.51
	MoFA	13	75.18	72.14
100	Lasso	48	76.98	73.77
	MoFA	45	77.04	73.93
200	Lasso	87	77.86	74.58
	MoFA	61	77.87	74.50
500	Lasso	187	78.93	75.22
	MoFA	85	78.68	74.81

Time-Worthiness				
K	Method	Groups	Train AUC	Test AUC
20	Lasso	19	69.29	69.72
	MoFA	15	69.11	69.69
100	Lasso	84	73.30	72.86
	MoFA	54	72.92	72.66
200	Lasso	84	74.22	73.76
	MoFA	76	73.74	73.44
500	Lasso	313	75.53	74.65
	MoFA	96	74.24	73.62

Table 1: Prediction results for true interest and time-worthiness tasks. Groups denotes the number of distinct feature groups in the selected subset.

operational alignment. While Lasso selects features purely based on statistical correlation, often choosing multiple features from different groups, MoFA explicitly considers group membership as part of its decision process, concentrating selections within fewer groups. This is particularly valuable in industrial settings where each additional feature group introduces dependencies on different data pipelines, engineering teams, and maintenance workflows.

4.3 Signal Pair Selection for Value Model

Background and Task. In large-scale recommendation systems, a *value model* (VM) is commonly used to estimate the overall value of user interactions. The VM serves as a key optimization signal, helping balance engagement, satisfaction, and monetization. In our current production system, the value model is defined as a linear combination of major behavioral signals, such as: $VM = a \cdot p_{\text{like}} + b \cdot p_{\text{click}}$, where p_{like} and p_{click} denote the predicted probabilities of user liking and clicking on a piece of content, respectively.

While this linear form is interpretable, it fails to capture interaction effects among behavioral signals, e.g., how simultaneous high values of both p_{like} and p_{click} may jointly reflect stronger user engagement. To better represent such dependencies, we consider extending the value model with higher-order interaction terms, such as: $VM' = a \cdot p_{\text{like}} + b \cdot p_{\text{click}} + c \cdot (p_{\text{like}} \times p_{\text{click}})$. However, enumerating all possible signal pairs is computationally expensive in online environments. Therefore, we employ MoFA to automatically identify the most promising interaction terms to include.

Feature Construction and Metadata. We construct a candidate pool $\mathcal{P} = \{(s_i, s_j)\}$ of signal pairs, where each pair represents a potential multiplicative interaction term. For each candidate pair (s_i, s_j) , we compute and record the following metadata: (1) the individual correlations between s_i and key engagement outcomes (e.g., user likes, saves, comments), (2) the pairwise correlation between $s_i \times s_j$ and these engagement signals, and (3) optional measures of signal reliability or coverage. These metadata fields are appended to the prompt as part of each candidate pair’s description, allowing the LLM to reason quantitatively about potential contribution and redundancy.

Online Experiment Results. From a pool of approximately 400 signal pairs, MoFA selected the top-10 feature interactions, as shown in Table 2. The three highest-ranked interactions: $p_{\text{external share}} \times p_{\text{profile tap}}$, $p_{\text{reshare button tap}} \times p_{\text{profile tap}}$, and $p_{\text{reshare button tap}} \times p_{\text{comment}}$ were sequentially incorporated into the value estimation model to assess their marginal contributions.

Rank	Feature Description
1	$p_{\text{external share}} \times p_{\text{profile tap}}$
2	$p_{\text{reshare button tap}} \times p_{\text{profile tap}}$
3	$p_{\text{reshare button tap}} \times p_{\text{comment}}$
4	$p_{\text{reshare button tap}} \times p_{\text{skip}}$
5	$p_{\text{external share}} \times p_{\text{skip}}$
6	$p_{\text{reshare button tap}} \times p_{\text{log time}}$
7	$p_{\text{reshare button tap}} \times p_{\text{skip}}$
8	$p_{\text{external share}} \times p_{\text{log time}}$
9	$p_{\text{comment surface enter}} \times p_{\text{external share}}$
10	$p_{\text{like}} \times p_{\text{reshare button tap}}$

Table 2: Top-10 feature interactions selected by MoFA. Each feature represents the product of two engagement signal probabilities. The top-3 features were used in the online experiment.

To evaluate the MoFA approach, we measured the total number of daily app sessions across users. For each new feature integration, we assessed the corresponding lift in the session metric relative to the baseline production recommendation system. The session improvements were 0.055%, 0.048%, and 0.018%, all of which are statistically significant. These evaluations validate the effectiveness of the MoFA feature selection process: By surfacing high-value feature pairs, our method advances holistic engagement optimization and provides a scalable path for the continual improvement of large-scale recommender systems.

4.4 Notification Behavior Prediction

Background and Task. In large-scale mobile applications, personalized notification delivery plays a key role in maintaining user engagement while avoiding overexposure or fatigue. Our production system employs a multi-task learning framework to predict user behavior across a wide range of notification-related outcomes. The downstream model simultaneously generates 38 prediction labels, including user click-through rate, dismissal rate, and other engagement signals. Among these, we primarily focus on improving two key metrics: the probability of **user click** and the probability

of **user dismissing notifications**, as these directly capture user interest and annoyance balance. The production model integrates multiple sub-networks and architectural modules, including multi-layer perceptrons, gating networks, and aggregation mechanisms, to jointly process heterogeneous feature types. The combination of such diverse inputs yields a feature space that is both large and heterogeneous, making manual or brute-force feature selection infeasible.

Feature Construction and Metadata. For this case, we construct a comprehensive pool of candidate features drawn from the production model. In our setup, we use MoFA to select 4,000 features from a total of 8,169 features. To effectively process this high-dimensional space, we partition the 8,169 features into $B = 10$ disjoint buckets, ensuring each prompt remains within the LLM’s context window. In Phase 1, the agent performs parallel reasoning to select 450 features from each bucket ($K' = 4,500$). In Phase 2, these candidates are merged into a refined pool where a final global selection pass is performed to reach the target of 4,000 features. Each feature is associated with metadata extracted from the currently deployed model. Specifically, for every candidate feature f_i , we include: its **feature category**, indicating whether it is a sparse, dense, or float feature. This metadata helps the LLM understand the modality of each feature. We also provide the LLM with the background of the task, including the model architecture and our main prediction objectives: user click probability and dismiss probability.

Experiment Results. To evaluate the effectiveness of MoFA in multi-task notification behavior prediction, we conducted offline experiments using normalized entropy (NE) loss as the primary metric on the test dataset. The current production model utilizes over 7,500 features, while our experiments randomly sampled 4,000 features for comparison. We report results for two key prediction tasks: probability of user click (p_{click}) and probability of user dismissing notifications (p_{dismiss}).

With 4,000 features selected by MoFA, the evaluation NE loss shows relative improvements over the production model (over 7,500 features). Specifically, p_{click} achieves a 0.065% NE win, and p_{dismiss} achieves a 0.198% NE win compared to the current production setup. Compared to a baseline where 4,000 features were selected via random shuffling, MoFA achieves a 0.19% NE win for p_{click} and a 0.332% NE win for p_{dismiss} . These results demonstrate that MoFA pipeline yields measurable improvements in notification behavior prediction, outperforming both the production model and random feature selection baselines.

5 Conclusion

In this work, we propose MoFA, an LLM-driven feature selection framework that integrates semantic understanding and quantitative metadata to enable context-aware and constraint-satisfying feature selection. Across real-world industrial cases, MoFA consistently improves downstream performance and operational efficiency: enhancing prediction accuracy and maintainability in user interest modeling, identifying meaningful interaction signals, and selecting high-utility feature sets for multi-task notification prediction.

These results demonstrate the practicality and versatility of leveraging LLM reasoning for intelligent feature selection in complex production environments.

6 Limitations

MoFA framework offers a sophisticated paradigm shift by treating feature selection as a sequential reasoning process. By integrating semantic descriptions with quantitative metadata, MoFA achieves high predictive utility while satisfying complex operational constraints. Several limitations remain.

First, because MoFA selects features one-by-one through iterative LLM prompting, the computational time scales linearly with the number of features K to be selected. This may introduce latency in systems requiring thousands of features. However, this is mitigated by the fact that feature selection is typically an offline preprocessing step, making reasoning time less critical than the resulting model efficiency.

Second, in large-scale industrial systems with thousands of candidates, providing a comprehensive list of all features and their metadata (e.g., importance scores, correlations) can challenge the input token limits of the LLM.

Finally, MoFA's sequential approach is inherently a greedy search, which might overlook optimal feature combinations that only show synergy when selected together.

We leave these refinements of current pipeline to the future work to broaden the applicability of MoFA pipeline.

Ethics and Privacy Statement

We utilize GPT-based models [26] solely for language polishing and proofreading. This work does not introduce any new datasets containing personal or sensitive information. Our agent relies on open-weight models, whose inherent biases or unsafe preferences may propagate to the system; such biases can further be amplified during inference. Practical deployment should therefore incorporate appropriate action validation, safety constraints, logging, and human oversight.

References

- [1] Avrim L Blum and Pat Langley. 1997. Selection of relevant features and examples in machine learning. *Artificial intelligence* 97, 1-2 (1997), 245–271.
- [2] Girish Chandrashekar and Ferat Sahin. 2014. A survey on feature selection methods. *Computers & electrical engineering* 40, 1 (2014), 16–28.
- [3] Gang Chen and Jin Chen. 2015. A novel wrapper method for feature selection and its applications. *Neurocomputing* 159 (2015), 219–226.
- [4] Marianne Cherrington, Fadi Thabtah, Joan Lu, and Qiang Xu. 2019. Feature selection: filter methods performance challenges. In *2019 International Conference on Computer and Information Sciences (ICCIS)*. IEEE, 1–4.
- [5] Kristy Choi, Chris Cundy, Sanjari Srivastava, and Stefano Ermon. 2022. Lm-priors: Pre-trained language models as task-specific priors. *arXiv preprint arXiv:2210.12530* (2022).
- [6] Manoranjan Dash, Kiseok Choi, Peter Scheuermann, and Huan Liu. 2002. Feature selection for clustering-a filter solution. In *2002 IEEE International Conference on Data Mining, 2002. Proceedings. IEEE*, 115–122.
- [7] Manoranjan Dash and Huan Liu. 1997. Feature selection for classification. *Intelligent data analysis* 1, 1-4 (1997), 131–156.
- [8] Chris Ding and Hanchuan Peng. 2005. Minimum redundancy feature selection from microarray gene expression data. *Journal of bioinformatics and computational biology* 3, 02 (2005), 185–205.
- [9] Magdalini Eirinaki, Jerry Gao, Iraklis Varlamis, and Konstantinos Tserpes. 2018. Recommender systems for large-scale social networks: A review of challenges and solutions. 413–418 pages.
- [10] Jean Feng and Noah Simon. 2017. Sparse-input neural networks for high-dimensional nonparametric regression and classification. *arXiv preprint arXiv:1711.07592* (2017).
- [11] Francesc J Ferri, Pavel Pudil, Mohamad Hatef, and Josef Kittler. 1994. Comparative study of techniques for large-scale feature selection. In *Machine intelligence and pattern recognition*. Vol. 16. Elsevier, 403–413.
- [12] Isabelle Guyon and André Elisseeff. 2003. An introduction to variable and feature selection. *Journal of machine learning research* 3, Mar (2003), 1157–1182.
- [13] Sungwon Han, Jinsung Yoon, Sercan O Arik, and Tomas Pfister. 2024. Large language models can automatically engineer features for few-shot tabular learning. *arXiv preprint arXiv:2404.09491* (2024).
- [14] Daniel P Jeong, Zachary C Lipton, and Pradeep Ravikumar. 2024. Llm-select: Feature selection with large language models. *arXiv preprint arXiv:2407.02694* (2024).
- [15] Ron Kohavi and George H John. 1997. Wrappers for feature subset selection. *Artificial intelligence* 97, 1-2 (1997), 273–324.
- [16] Vipin Kumar and Sonajharia Minz. 2014. Feature selection. *SmartCR* 4, 3 (2014), 211–229.
- [17] Cosmin Lazar, Jonatan Taminiau, Stijn Meganck, David Steenhoff, Alain Coletta, Colin Molter, Virginie De Schaetzen, Robin Duque, Hugues Bersini, and Ann Nowe. 2012. A survey on filter techniques for feature selection in gene expression microarray analysis. *IEEE/ACM transactions on computational biology and bioinformatics* 9, 4 (2012), 1106–1119.
- [18] Ismael Lemhadri, Feng Ruan, Louis Abraham, and Robert Tibshirani. 2021. Lasonet: A neural network with feature sparsity. *Journal of Machine Learning Research* 22, 127 (2021), 1–29.
- [19] David D Lewis. 1992. Feature selection and feature extraction for text categorization. In *Speech and Natural Language: Proceedings of a Workshop Held at Harriman, New York, February 23-26, 1992*.
- [20] Dawei Li, Zhen Tan, and Huan Liu. 2025. Exploring large language models for feature selection: A data-centric perspective. *ACM SIGKDD Explorations Newsletter* 26, 2 (2025), 44–53.
- [21] Jundong Li, Kewei Cheng, Suhang Wang, Fred Morstatter, Robert P Trevino, Jiliang Tang, and Huan Liu. 2017. Feature selection: A data perspective. *ACM computing surveys (CSUR)* 50, 6 (2017), 1–45.
- [22] Jianhao Li and Xianchao Xiu. 2025. LLM4FS: Leveraging Large Language Models for Feature Selection and How to Improve It. *arXiv preprint arXiv:2503.24157* (2025).
- [23] Zongxia Li, Andrew Mao, Daniel Stephens, Pranav Goel, Emily Walpole, Alden Dima, Juan Fung, and Jordan Boyd-Graber. 2024. Improving the TENOR of Labeling: Re-evaluating Topic Models for Content Analysis. *arXiv:2401.16348 [cs.CL]*
- [24] Shan Luo and Zehua Chen. 2014. Sequential Lasso cum EBIC for feature selection with ultra-high dimensional feature space. *J. Amer. Statist. Assoc.* 109, 507 (2014), 1229–1240.
- [25] AI Meta. 2025. The llama 4 herd: The beginning of a new era of natively multimodal ai innovation. <https://ai.meta.com/blog/llama-4-multimodal-intelligence/>, checked on 4, 7 (2025), 2025.
- [26] OpenAI. 2023. GPT-4 Technical Report. *arXiv:2303.08774 [cs.CL]*
- [27] Jiliang Tang, Salem Alelyani, and Huan Liu. 2014. Feature selection for classification: A review. *Data classification: Algorithms and applications* (2014), 37.
- [28] Robert Tibshirani. 1996. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society Series B: Statistical Methodology* 58, 1 (1996), 267–288.
- [29] Özge Uncu and IB Türkşen. 2007. A novel feature selection approach: combining feature wrappers and filters. *Information Sciences* 177, 2 (2007), 449–466.
- [30] Yutaro Yamada, Ofir Lindenbaum, Sahand Negahban, and Yuval Kluger. 2020. Feature selection using stochastic gates. In *International conference on machine learning*. PMLR, 10648–10659.
- [31] Tianze Yang, Tianyi Yang, Fuyuan Lyu, Shaoshan Liu, et al. 2024. Ice-search: A language model-driven feature selection approach. *arXiv preprint arXiv:2402.18609* (2024).
- [32] Ming Yuan and Yi Lin. 2006. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society Series B: Statistical Methodology* 68, 1 (2006), 49–67.