Interpreting Cross-Domain User Activity Sequences with LLMs

Anonymous submission

Abstract

Sequential or time-stamped activity logs provide objective records of digital application usage, yet their granularity and noise often obscure meaningful insights into people's work. Such insights are essential for improving digital products in ways grounded in real-world user interactions. Prior research has applied deep learning models to cluster user actions into high-level workflow activities, but these approaches are highly sensitive to noise and struggle to generalize across applications. To address this limitation, we introduce WorkflowView, a framework that leverages large language models (LLMs) to abstract low-level action sequences into high-level workflow activities. We establish the effectiveness and generality of our approach across three distinct, challenging sequential tasks and diverse domains: (a) zero-shot task description reconstruction from browser logs (achieving high semantic similarity, $\mu_{sim} = 0.91$), (b) few-shot student dropout prediction using MOOC activity logs (reaching weighted $F_1 = 0.90$ with only five few-shot examples), and (c) anonymized, privacy-preserving analysis of AI tool integration within document workflows on a proprietary platform. Our work demonstrates that LLM-based abstraction is a robust and efficient path forward for transforming low-level behavioral data into high-level, interpretable, and actionable insights. We also discuss practical considerations for deploying LLM-based inferences within logging infrastructures, including computational efficiency and user privacy.

1 Introduction

Terabytes of user interface (UI) activity logs are captured every hour as users interact with digital applications. These logs enable unobtrusive analysis of usage patterns, facilitate bug identification, and support iterative deployment of product improvements that better align with users' needs. UI activity logs provide an objective account of what actions users perform and when they perform them (e.g., (`DD/MM/YY HH:MM:SS`, `ClickedLayoutRibbon`)). However, such timestamped action sequences are often too granular and noisy to yield a clear view of the high-level task a user is performing within the application. A single high-level task (e.g., formatting the content of a document) may comprise hundreds of actions executed over a 10-15 minute interval, making the activity sequence highly *granular*. Moreover, these sequences may include actions that are not directly related to the user's underlying intent, introducing *noise*. For example, when users briefly click on unrelated features to intentionally or unintentionally explore the interface.

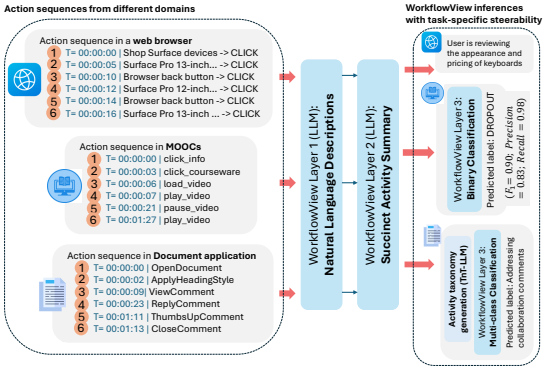

Figure 1: We propose an LLM-based framework for hierarchical abstraction of user activity sequences into interpretable high-level activities (WorkflowView). The left panel illustrates raw action sequences from three domains. WorkflowView enables downstream inferences with task-specific steerability, such as reconstructing user intent in browsers, predicting student dropout in MOOCs, and privacy-preserving categorization of document-centric workflows.

Earlier studies that model time-stamped activity logs to understand user behavior have relied on statistical techniques such as frequent itemset mining and sequential pattern mining (Mannila et al., 1997; Cuke et al., 2009; Agrawal and Srikant, 1995; Agrawal et al., 1993). These approaches have been noted to struggle with incorporating domain context and with explicitly modeling noise in user behavior (Dev and Liu, 2017). More recent work has explored adapting language modeling techniques to sequential log data; for example, using LSTMs to preemptively identify when users might need assistance within an application (Nambhi et al., 2019), or training BERT- and LLM-based classifiers to detect anomalies in logs (Guo et al., 2021; Zhou et al., 2024). While these methods demonstrate the promise of interpreting activity logs using lan-

guage models, they typically operate in settings that require task-specific fine-tuning on thousands of annotated training samples.

Motivated by the strong generalization capabilities of large language models (LLMs) across tasks and domains, this work investigates whether state-of-the-art LLMs can interpret real-world timestamped action sequences that do not follow the usual syntax or semantics of natural language and infer the high-level activities that users perform as part of their workflows. LLMs are also known to integrate presented instances with broader encoded knowledge (Bai et al., 2024), which may further enrich observability of system and user states. To this end, we propose WorkflowView, a hierarchical abstraction of granular action sequences using LLMs. In WorkflowView, the initial layer generates natural language descriptions of observed activity, while subsequent layers infer higher-level activities and, optionally, categorize them into a set of discovered or predefined categories. To demonstrate the generality of the proposed approach, we evaluate WorkflowView across three domains that differ in action set cardinality and in the degree to which user behaviors are mutually exclusive. An overview of the method is shown in Figure 1.

Our results show that WorkflowView provides a reliable abstraction over action sequences across diverse tasks and domains. Specifically, we find that the method *(a)* generates activity descriptions that closely align with ground-truth tasks performed in a browser (e.g., prediction: the user is trying to "find a car while sorting by lowest price"; ground truth: the user wants to "find the cheapest car"), *(b)* predicts student dropout in MOOCs with a weighted F_1 score of 0.90 while using only five in-context examples (a performance comparable to several state-of-the-art predictive models trained on thousands of labeled instances), and *(c)* contextualizes the use of AI tools in a proprietary document creation, collaboration, and consumption application by interpreting action sequences, discovering task categories, and performing multi-class classification. We further show that such anonymous, privacy-preserving, and aggregated insights can inform user-centric product improvements.

Because WorkflowView relies on LLM-based inference over action sequences, we discuss practical considerations around deployment, including cost, latency, and user-privacy, as well as the limitations of our approach. We also outline a broader vision in which LLM capabilities are embedded deeper into the logging infrastructure. This vision is especially relevant in the context of human–AI collaboration, while maintaining strong guarantees around user privacy and security.

2 Related work

Below, we categorize and discuss related work into three themes: *(a)* modeling activity logs, *(b)* discovering user intents from user utterances, and *(c)* using LLMs to model non-language data.

Modeling activity logs: Prior work on interpreting timestamped UI logs has largely framed the problem as pattern mining or sequence modeling. Techniques such as frequent itemset mining and sequential pattern mining have been widely used to extract common action patterns from large log corpora (e.g., identifying frequently occurring operation groups) (Mannila et al., 1997; Cuke et al., 2009; Agrawal and Srikant, 1995; Agrawal et al., 1993). While effective at identifying recurring structures, these statistical approaches are largely domain-agnostic: they treat UI actions as abstract tokens without semantic grounding (Dev and Liu, 2017) and are sensitive to noise and spurious correlations in action sequences (Yang et al., 2002). Subsequent work addressed some of these limitations through learning-based approaches, including RNN/LSTM- and Transformer-based models (Hochreiter and Schmidhuber, 1997; Vaswani et al., 2017), applied to domain- and task-specific applications (Nambhi et al., 2019; Krishna et al., 2018; Zhu et al., 2021). However, these methods rely on task-specific training data and hand-crafted labels, making them costly to deploy across new domains, tasks, or evolving user behaviors. In contrast, WorkflowView relies on LLM-based inference via prompting, enabling flexible adaptation across tasks and domains without fine-tuning or annotating data, while explicitly abstracting away low-level noise through hierarchical reasoning.

Intent discovery over user utterances: A related line of work focuses on inferring user intent from textual interactions, such as search queries (Wang et al., 2022) or conversational utterances in dialogue systems (Schuurmans and Frasincar, 2019). Modern dialogue systems and virtual assistants typically include an intent classification module that maps user input to predefined task labels (e.g., booking a flight or checking the weather), often trained using supervised learning over large annotated corpora (Serban et al., 2015). More recent

work explores discovering new or evolving intents by clustering user queries that fall outside known categories (Shah et al., 2025; Wan et al., 2024). A key distinction between this body of work and ours lies in the nature of the input: textual utterances are already semantic and human-interpretable, and often explicitly encode user goals (e.g., "find the cheapest car" or "schedule a meeting"). In contrast, our work operates on telemetry data consisting of low-level UI events, where intent must be inferred indirectly from noisy, granular action sequences. This setting is both more challenging and more ubiquitous in modern applications, motivating the need for methods that can bridge raw interaction logs and high-level intent.

LLMs for non-language sequential data: Beyond text, recent work has examined the ability of LLMs to reason over non-language data. Existing approaches include learning projection layers to map image or numeric sensor data into representations suitable for LLM-based inference (Verma et al., 2024; Moon et al., 2024), adapting LLM embeddings for time-series classification (Kaur et al., 2025), and reprogramming time series into textual prototype representations that align more naturally with LLM pretraining (Jin et al., 2024). Liu et al. 2024a demonstrate that off-the-shelf LLMs such as GPT-4 (Achiam et al., 2023) can outperform pretrained zero-shot baselines (and, in many cases, supervised models) on forecasting numeric sequences across domains including epidemiology, finance, and weather. These results suggest that LLMs can, to some extent, interpret structured sequences with limited linguistic content by leveraging patterns learned during large-scale pretraining. Building on this insight, WorkflowView extends zero-shot and few-shot LLM prompting to the domain of user activity logs.

3 WorkflowView: Hierarchical abstraction of activity logs with LLMs

WorkflowView is a simple yet effective framework that leverages large language models (LLMs) to reason over activity sequences. The method demonstrates that LLMs can be prompted to address a range of sequence modeling tasks across domains in zero-shot or few-shot settings, highlighting ease of customization without fine-tuning. To encourage stage-wise abstraction from low-level actions to high-level intents, WorkflowView adopts a hierarchical design. Specifically, activity sequences are first converted into detailed natural language descriptions (Layer 1), after which the high-level activity captured by these descriptions is inferred (Layer 2). If required by the task, additional layers can be introduced to further categorize the inferred high-level activity into known or discovered classes—for example, predicting student dropout in a MOOC or distinguishing between active document editing and text formatting. Figure 1 provides an overview of the approach along with example outputs from the datasets used in this work.

The hierarchical LLM-based inference is motivated by two principles: *modularity* and *progressive denoising*. Modularity ensures that the outputs of lower layers (i.e., action sequence $\rightarrow$ natural language description $\rightarrow$ high-level task inference) can be reused across multiple objectives (such as frequent task discovery at the population level or categorization of individual sequences) by adapting only the higher layers. Progressive denoising is essential for modeling action sequences with LLMs, as it enables the transformation of raw timestamped actions into coherent textual representations that are better suited for higher-order reasoning. For instance, lower layers may capture temporal patterns in natural language, such as "the user responded to a collaborator's comment after no significant activity for N minutes." In this case, low-salience actions are denoised at earlier layers, and depending on the value of N (e.g., 2 *vs.* 10 minutes), subsequent layers can characterize the level of deliberation involved in responding to the comment. See our discussion on the effectiveness of progressive denoising in Appendix A.2.

We provide the prompts used in WorkflowView in Appendix Tables 7, 8, 9, and 10 to support reproducibility and future work. In the following section, we evaluate WorkflowView on three tasks spanning three domains: inferring browser tasks, predicting student dropout in MOOCs, and contextualizing the use of an AI tool in a proprietary document application. Given the substantial variation in action spaces and behavioral patterns across these domains, our experiments are designed to evaluate WorkflowView's effectiveness and generalizability.

4 Applications & Evaluation of WorkflowView

4.1 Inferring tasks from browser action logs

Task and dataset: We evaluate the ability of WorkflowView to infer the tasks people do on browsers using observed action logs alone. We use the Mind2Web dataset (Deng et al., 2023), which con-

Metric	Global	Website
MRR	0.90 ($\pm$0.08)	0.94 ($\pm$0.06)
Recall@1	0.86 ($\pm$0.13)	0.92 ($\pm$0.09)
Recall@3	0.94 ($\pm$0.07)	0.98 ($\pm$0.04)
Recall@5	0.96 ($\pm$0.05)	0.99 ($\pm$0.03)
Recall@10	0.98 ($\pm$0.01)	0.99 ($\pm$0.01)

Table 1: Embedding-based retrieval of ground-truth task descriptions using task descriptions generated using WorkflowView; the candidate set varies as 'global' or 'website-specific' across the two settings. $\mu(\pm\sigma)$

.

tains an ordered sequence of web actions taken in a browser to complete 2,022 general-purpose web tasks described in natural language. The tasks span 137 websites and 5 different domains: service (e.g., *gov.uk*), shopping (e.g., *instacart.com*), entertainment (e.g., *espn.com*), travel (e.g., *delta.com*), and information (e.g., *finance.yahoo.com*). The action space for this dataset is characterized by HTML UI elements (for instance, [button] 'Go Back', [textbox] 'Enter your name') that the user interacts with on a webpage and the operation they perform (like CLICK, TYPE, or SCROLL). Our goal is to perform LLM abstractions over the action sequences, as exemplified in Table 2 (action sequences), to predict the task the users are doing across different websites (task). Methodologically, for this task, we operationalize WorkflowView using the prompts shown in the Appendix, where Layer 1 (shown in Table 7) provides a detailed description of the activity sequences in natural language and Layer 2 (shown in Table 8) infers the overall task the user is doing and generates its succinct description. It is worth noting that this evaluation is a zero-shot setting. All our key results are based on experimentation with GPT-4o (more specifically `gpt-4o-2024-05-13`), a leading proprietary state-of-the-art large language model released by OpenAI (OpenAI, 2024). However, we also demonstrate that WorkflowView works effectively with smaller & open-weights models like Phi-4 (Abdin et al., 2024) and `gpt-oss-20b` (OpenAI, 2025) in App Table 6.

Evaluation settings: As the first measure to compare the generated task descriptions from sequences of web interactions and their corresponding ground-truth descriptions, we compute the cosine similarity between the embeddings of the descriptions obtained from the `text-embedding-ada-002` model (OpenAI, 2022). Additionally, we compute retrieval metrics like Mean Reciprocal Rank (MRR) and Recall@K

Action Sequence	[svg] → CLICK, [link] Your lists → CLICK, [link] Create a list → CLICK, [svg] → CLICK, [span] Walgreens → CLICK, [textbox] Add a title (Required) → TYPE: Walgreens, [img] → CLICK, [button] Next → CLICK, [link] Personal Care → CLICK, [svg] → CLICK, [img] → CLICK, [span] Add to list → CLICK, [checkbox] Walgreens New → CLICK, [button] Done → CLICK, [path] → CLICK, [path] → CLICK, [path] → CLICK, [svg] → CLICK, [img] → CLICK, [span] Add to list → CLICK, [checkbox] Walgreens New → CLICK, [button] Done → CLICK, [path] → CLICK, [link] View More → CLICK, [img] → CLICK, [span] Add to list → CLICK, [checkbox] Walgreens New → CLICK, [button] Done → CLICK, [button] Back → CLICK, [path] → CLICK, [link] Shower Essentials → CLICK, [img] → CLICK, [span] Add to list → CLICK, [checkbox] Walgreens New → CLICK, [button] Done → CLICK, [button] Back → CLICK, [link] Lists → CLICK
Generated Task Description	Create a Walgreens shopping list and add personal care and shower essentials items.
Ground-truth Task Description	Create a new list and add four items from the personal care category at Walgreens.

Table 2: Qualitative example of task description generated using WorkflowView using the action sequence, and the corresponding ground-truth task description.

($K \in \{1, 3, 5, 10\}$) under two settings. In the first setting (i.e., 'global'), we retrieve the most similar ground-truth task description across the entire dataset for each of the generated task descriptions; whereas, in the second setting (i.e., 'website-specific'), we retrieve the most similar ground-truth description across a website for each of the generated descriptions belonging to the same website.

Results: The average similarity (and standard deviation) between the generated and ground-truth tasks is 0.911 ($\pm$ 0.042); $N = 2,022$ tasks. The notably high absolute similarity scores indicate the accurate inferences made using WorkflowView. Table 1 shows the average MRR and Recall@K values (along with standard deviations). The near-perfect MRR and Recall@K values also indicate that the true ground-truth descriptions are ranked at the top for a large majority of the corresponding generated task descriptions. In Table 2, we qualitatively illustrate the close alignment between a generated task description and the corresponding ground-truth description; an expanded set of qualitative examples is present in Appendix Table 11. The key strength of our work lies in demonstrating zero-shot, cross-domain generality of WorkflowView; nonetheless, we compare against domain-specific fine-tuned seq2seq (Sutskever et al., 2014) baselines for this task in Appendix A.1.1.

4.2 Predicting student dropouts based on MOOC interaction logs

Task and dataset: To assess the method's generalizability across diverse tasks and domains, we

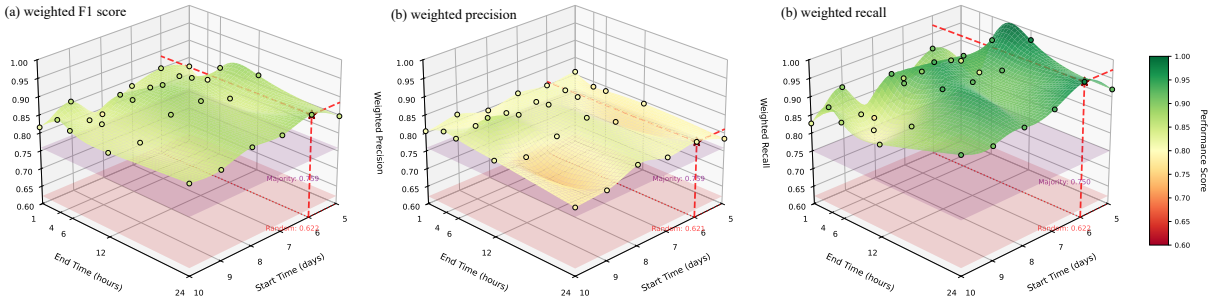

Figure 2: Dropout prediction performance using Workflowview. The plots show the weighted F_1 score, precision, and recall in response to the variations in 'start time' (i.e., time when the action sequences are modeled) and 'end time' (i.e., time before the last action until which the action sequences are modeled). For comparison, we include the scores corresponding to baselines where only the 'majority' class would be predicted (i.e., all dropout) and predictions based on biased 'random' guesses as per prior class probabilities. The best F_1 score (F_1 = 0.89; Precision = 0.81; Recall = 0.97) correspond to a start time of 6 days and an end time of 24 hours. The number of few-shot examples provided to WorkflowView were 3 for this analysis; Figure 3 below shows the sensitivity to the number of few-shot examples.

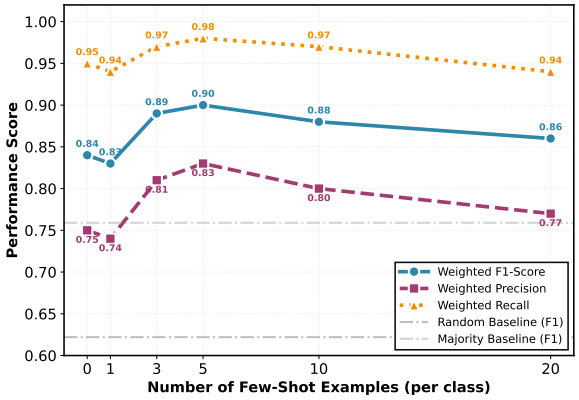

Figure 3: Variation in predictive performance of WorkflowView (weighted F_1, precision, and recall) on the MOOC dropout prediction task, in response to the number of few-shot examples considered ($N \in \{0, 1, 3, 5, 10, 20\}$).

experiment with interaction logs of a MOOC software to predict student dropouts. The test set of the dataset curated by Feng et al. (2019) comprises interaction logs from a total of 44,008 unique students enrolled in 247 unique courses, resulting in 67,699 unique (student, course) pairs. Of the 67,699 unique enrollments, 51,316 (75.8%) resulted in a dropout. 22 unique actions were logged from all the student interactions, representing the action space. The goal of this task is to process the time-stamped sequence of actions at least N hours *before the last action* using WorkflowView to determine if the enrollment is going to result in a dropout, such that $N \in \{1, 6, 12, 18, 24\}$ hours. The design for this predictive task takes into account a potential intervention to take place when the student performs their currently last action that may discourage them from dropping out.

Adapting WorkflowView: For this task, to facilitate a binary classification, we adapt WorkflowView to have a third categorization layer on top of the first two layers (natural language description and succinct summary). Effectively, if the final binary classification labels are accurate, it indicates that WorkflowView can interpret and extract meaningful task-specific signals from the raw action sequences. This particular task and the low-barrier adaptation of WorkflowView to address it emphasizes the modularity of the underlying hierarchical abstractions. We also explore WorkflowView's compatibility to few-shot settings, by modifying the prompts at each layer to include illustrative mappings. Specifically, for the natural language description layer (Layer 1) this was done by providing the mapping between action sequences and the final category; for the succinct summary layer (Layer 2) this was done by additionally including the natural language descriptions from the previous layer for the same examples, and similarly, for Layer 3, we additionally included the succinct summaries for the examples. The prompts used to adapt WorkflowView for this task are shown in the Appendix Tables 9 and 10. We explored few-shot settings where the number of examples per category varied in $\{1, 3, 5, 10\}$, where that many examples were randomly sampled from the training set per category (i.e., the total number of examples were twice as many).

Evaluation setup: The evaluation is designed to measure how effectively and reliably can WorkflowView perform the binary classification task of predicting student dropouts from MOOC action sequences. Our evaluations are centered around two axes that could precipitate predictive variability: *(a)* time horizon for the action sequences under consideration, and *(b)* the number of few-shot examples provided to the model. For the former, we vary the start time as well as the end time and observe the weighted F_1 score, precision, and recall for each combination. For the latter, we explore both zero-shot and few-shot settings, while varying the number of examples considered in the few-shot setting. In the few-shot setting, we randomly sam-

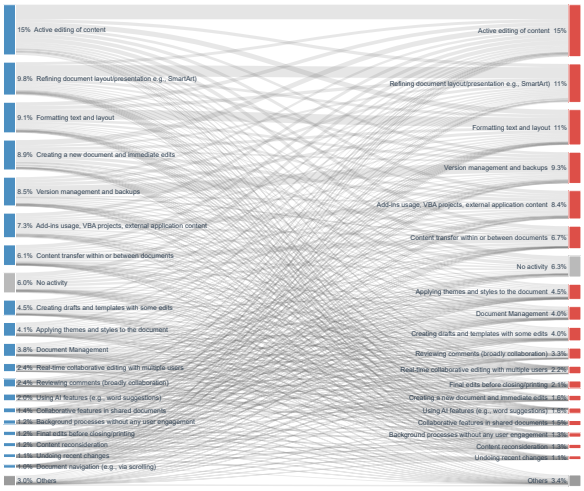

Figure 4: Activities users do in the context of document editing, at most 30 minutes before (left, in blue) and after (right, in red) prompting the integrated AI tool and accepting its output. The action sequences are processed using WorkFlowView to discover the high-level categories; corresponding definitions are provided in the adjoining table.

- **Active editing of content**: Modifying the content of a document, such as copying, pasting, deleting, and reorganizing text.
- **Refining document layout/presentation**: Refining and enhancing document layout and presentation, including inserting graphics.
- **Formatting text and layout**: Changing the appearance of text and its layout, including font adjustments, applying styles, and using tools like Format Painter.
- **Add-ins usage, VBA projects, external application content**: Interactions with external applications or add-ins within the context of document editing.
- **Version management and backups**: Create different versions or backups of a document.
- **Creating a new document and immediate edits**: Starting a new document and immediately engaging in editing or content insertion.
- **Content transfer within or between documents**: Copying and pasting content within or between documents.
- **Applying themes and styles to the document**: Organizing and structuring document content, including applying themes and styles.
- **Creating drafts and templates with some edits**: Creating drafts or templates, making edits, and possibly using draft generation features.
- **Reviewing comments (collaboration)**: Engaging with comments, indicating a review or collaboration phase.
- **Real-time collaborative editing with multiple users**: Engaging in real-time collaboration and editing with others.
- **Final edits before closing/printing**: Preparing the document for presentation or distribution, including final edits, formatting, and printing.
- **Collaborative features in shared documents**: Shared document activities, including co-authoring, managing comments, and using collaborative features.
- **Using AI features (e.g., word suggestions)**: Using AI features to edit the document content.
- **Content reconsideration**: Experimenting with content by adding and then removing it, indicating reconsideration of content placement or inclusion.
- **Document navigation**: Navigating through the document, including moving the cursor or scrolling.

Table 3: Discovered document editing categories and their corresponding descriptions.

ple K examples per category from the train set of the data; we acknowledge that better sampling strategies are possible (Wang et al., 2020). To limit the amount of experimentation we first evaluate the performance of WorkflowView on a 2-dimensional hyper-parameter grid of start and end times while fixing the number of few-shot examples to 3, and then evaluate the sensitivity to the number of few-shot examples.

Results: Figure 2 shows the predictive performance in response to variations in start and end time hyper-parameters. For reference, we also include comparisons with two random baselines: 'majority', where only the majority class is predicted (i.e., all sequences are categorized as dropout) and 'random', where the categorizations are based on class probabilities based on the training data distributions. The first key observation is that regardless of the start and end times hyperparameters, WorkflowView categorizations are consistently and notably better than either of the baselines. In fact, the best performance across all the combinations comes out at an F_1 score of 0.89 (Precision $= 0.81$ and Recall $= 0.97$) corresponding to a start time of 6 days and an end time of 24 hours before the last activity. It is worth noting that this performance is on par with several learning-based methods that utilize over hundreds of thousands of training examples. [1] Next, we fix the start time of the activity sequence used for predictive modeling to 6 days and the end time to 24 hours, and vary the number of few-shot examples supplied to the model. Figure 3 shows that using 3 or 5 few-shot examples per category improves the performance considerably over the zero-shot setting (F_1 improves from 0.84 to 0.89 and 0.90, respectively). The minor drop in performance with only a single few-shot example can be explained by the two operating modes of in-context learning (Lin and Lee, 2024; Min et al., 2022): with insufficient demonstrations, models tend to rely on retrieving familiar tasks from pretraining ('task retrieval') rather than adapting to the presented task ('task learning'). However, on further increasing the number of in-context learning examples from $5 \rightarrow 10 \rightarrow 20$, the performance drops possibly because of increasing context length. More broadly, applying WorkflowView to predict student dropouts based on MOOC interaction logs not only indicates the effectiveness of the method regardless of the hyperparameters related to sequence length and duration but also provides insights into how the method interacts with the broader literature around in-context learning.

4.3 Analyzing before/after activities around key actions: A case-study of how AI tools help in document workflows

As AI assistance tools are transforming how users engage with documents across different digital applications, we conduct a case-study on analyzing the action sequences before and after users accept the response provided by an AI tool (as captured by

[1] Fu et al. (2021) train a long short-term memory network to build a predictive model that achieves and F_1 score of 0.869 on the binary classification task of MOOC dropout prediction. Similarly, Basnet et al. (2022) propose training-based approaches that rely on large-scale annotated data and result in an F_1 score of 0.84; Feng et al. (2019) report an F_1 score of 0.91. See App. A.1.2 for baseline comparisons.

a specific action in the application telemetry). The AI tool considered in this study is embedded as a product feature within a proprietary document editing application with hundreds of millions of active users. The application allows the users to prompt the AI tool at any stage of their workflow while working with a document and accept or discard its presented output. The case-study demonstrates how WorkflowView could provide interpretable and actionable user-centric insights that could improve interactions and product design.

We used WorkflowView to analyze the anonymous and privacy-preserving telemetry of users of the proprietary document application. We sampled $50,000$ users who had interacted with the AI tool at least once in the month of June 2025 and were located in the United States with their application language set to `us-en`. Users consented to log collection as part of the user agreement. Note that the interaction logs are devoid of textual data and writer data, and the telemetry that captures users' interactions with the application UI is highly granular and include approximately 2000 unique actions. We only present aggregated, percentage-based insights over the random sample of the users.

Two key modifications exist beyond applying Layers 1 (i.e., natural language descriptions) and 2 (succinct activity summary) of WorkflowView for this task. Since there is no list of prior activities that the action sequences need to be mapped to, these activities have to be 'discovered' from the data. Once these activities have been discovered, there is a need for a categorization layer that maps the succinct summaries to one of the discovered categories. For the category discovery step, we use an existing method (TnT-LLM (Wan et al., 2024)) that does end-to-end label generation based on the raw succinct activity summaries (i.e., Layer 2's output). The identified labels are then used in Layer 3 of WorkflowView for multi-class classification, akin to the binary classification task for MOOC dropout prediction. We included the description of the high-level categories that were obtained using TnT-LLM in the previous step to inform the multi-class classification. This case-study also illustrates another example of easy adaptation of WorkflowView to applications that may involve identifying categories of activities or model evolving user behavior, where existing categories may become outdated over time.

Dataset: We processed the sequences before and after (at most 30 minutes in duration) each of the occurrences of the action that indicates keeping AI tool's output using WorkflowView. Finally, for discovering the category of activities using TnT-LLM, we used 20% of the occurrences and then inferred the categories (using Layer 3) on the entire action sequence set.

Analysis and Insights: Figure 4 shows that active content editing (described as activities involving modifying content of a document, such as copying, pasting, deleting, and reorganizing text in Table 3) is the most frequent activity both before (15%) as well as after (15%) AI assistance, indicating its prominence in document-related workflows where AI tools are used. Active editing of content tends to continue as such after accepting AI tool's outputs or, in certain cases, the user tends to transition to other activities like formatting text and its layout or transferring content within or across documents. It is worth noting that the share of activities pertaining to formatting or refining layouts is greater *after* the AI tool's responses are accepted when compared to their share before, which may indicate that users try to incorporate the AI tool's output in a manner that is consistent with the original content's formatting. The insights obtained with WorkflowView enable interpreting user engagement patterns from noisy and granular action sequences. Additionally, these insights also offer actionable guidance for product improvements such as introducing more context-aware formatting suggestions or adaptive layout tools that align with post-AI interaction behaviors.

It is also worth noting that in an evolving landscape where AI tools are changing how users interact with applications, it is a strength that the activities (and the corresponding descriptions) are *synthesized* with the activity summaries inferred by WorkflowView from the action sequence data, rather than being predefined labels. This ensures that the taxonomy reflects actual user behavior as it evolves over time. We discuss practical considerations around efficiency in the context of real-world deployment in the following section.

5 Discussion: Deployment and Extensions

WorkflowView is an LLM-powered approach to do hierarchical abstractions over activity sequences to understand users' behavior within digital applications. We demonstrate that the method can be easily adapted to work with a diverse set of tasks involving action sequences (task description generation, binary classification, category discovery and

multi-class classification) across different domains (browser, MOOC, and document editing application). Quantitative and qualitative analyses demonstrate that WorkflowView is on par with training-based models for these tasks in zero-shot or few-shot settings. The results indicate the promise of embedding LLM-powered inferences in the lowest level of data infrastructure to drive user-centric product improvements. Below, we discuss some of the future extensions and applications of WorkflowView.

Deployment cost and latency: Relying on LLM-based inference requires careful consideration of deployment cost and latency, particularly for applications with large user bases where activity logs may span terabytes of data. Two trends are worth noting. First, the cost and latency of LLM inference have decreased rapidly in recent years (Agarwal et al., 2023; Cottier et al., 2025). In parallel, there has been growing support for deploying smaller language models, which our evaluations suggest can perform on par with larger models for activity inference tasks. For example, Phi-4 (14B parameters; (Abdin et al., 2024)) achieves performance comparable to GPT-4o on the web browser inference task while requiring substantially fewer hardware resources (Appendix Table 6). Together, these trends make it increasingly feasible to apply LLM-based methods such as WorkflowView to large-scale activity logs. In the near term, WorkflowView can be deployed in offline settings to help developers understand how users interact with their applications and to identify opportunities for product improvement. Periodic offline analyses can also surface shifts in user behavior over time, which is particularly relevant in dynamic human–AI collaborative workflows. Such deployments can further control cost by operating on representative samples of users, while avoiding the latency constraints associated with real-time inference.

Multimodal extensions of WorkflowView: In this work, WorkflowView operates on activity sequences captured in text. However, given the multimodal capabilities of modern LLMs, the framework can be naturally extended to incorporate UI screenshots captured at key transition points during user interaction. Visual context can provide complementary signals about user behavior, reduce reliance on application-specific instrumentation, and ground textual descriptions in the actual interface state. Figure 5 qualitatively illustrates such a multimodal setting, where both high-level task inference and behavioral descriptions are captured accurately.

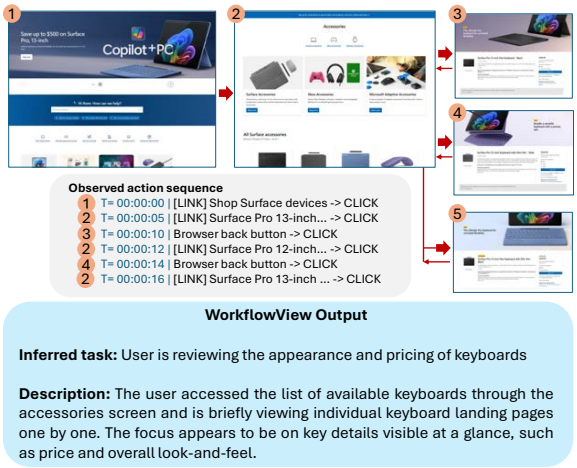

Figure 5: Illustrative multimodal extension of WorkflowView for task inference using browser snapshots and UI click logs. In this example, task inference is performed unobtrusively as the user interacts with the application. The visual modality complements the textual activity sequence by grounding descriptions in on-screen context (e.g., visible content and time spent), leading to more accurate and informative inferences than action sequences alone.

Multimodal extensions of WorkflowView open the door to real-time, proactive AI assistance that supports users in achieving their high-level goals. For example, a user browsing an e-commerce website could be offered structured recommendations—such as items to compare, stratified by viewed and unviewed—based on their inferred goal (e.g., evaluating alternative keyboards) and current progress toward that goal. More broadly, accurate real-time modeling of user behavior, encompassing both intent and task progress, is likely to be foundational for human–AI collaboration, enabling seamless hand-offs between users and AI systems.

6 Conclusion

Building on the generalization capabilities of LLMs, including their demonstrated effectiveness on non-language data, we introduce WorkflowView, a framework for hierarchical abstraction over action sequences to infer users' high-level activities within digital applications. We show that WorkflowView can be applied reliably across three diverse domains and across multiple tasks. Through a case study on real-world telemetry from a proprietary application, we illustrate how WorkflowView enables anonymous, privacy-preserving, and aggregated analysis of user behavior that can inform user-centric product improvements. Finally, we outline multimodal extensions of WorkflowView that can support more effective human–AI collaboration and discuss key deployment considerations.

7 Limitations

Privacy and security considerations: It is critical to consider the privacy and security implications associated with the design, deployment, and future extensions of WorkflowView. First, UI action sequences should only be collected with informed user consent, and inferences should be limited to behavioral understanding that does not reveal PII or sensitive content—for example, inferring that a user is "actively applying formatting changes to text" rather than "actively formatting text in a legal contract." For real-time assistance, particularly in multimodal settings where private data could appear in UI screenshots, operationalization could involve strictly performing *on-device* inferences, while logging only privacy- and security-compliant textual abstractions server-side for offline analysis that informs product improvements. Concrete privacy budgets or differential privacy (DP)-style guarantees could be explored in future work. Transparent and informed user consent is essential to ensure trust in AI-powered technologies.

Limitations: First, the action names that make up activity sequences must convey meaningful information about user interactions (e.g., 'ClickLayoutRibbon' rather than 'Action1'). This limitation also highlights the importance of developing an informative logging infrastructure to fully leverage LLM capabilities. Second, future work could explore more token-efficient prompting mechanisms to represent raw activity sequences, as our current approach uses a direct textual representation of time-stamped actions. Simple strategies, such as chunking actions based on temporal proximity, could substantially reduce token counts. Finally, while this work evaluates off-the-shelf LLMs under zero-shot and few-shot settings with a focus on cross-task and cross-domain generalizability, future research could investigate large-scale pre-training on action sequences from diverse domains to further improve generalization across both in-sample and out-of-sample tasks.

Data and code: Two of the datasets were curated by prior work are publicly available (Deng et al., 2023; Feng et al., 2019); we comply with their terms of use. Users of the proprietary document application consented to activity log collection as part of the user agreement. We will release the code associated with experiments on the publicly available datasets to facilitate reproducibility.

References

Marah Abdin, Jyoti Aneja, Harkirat Behl, Sébastien Bubeck, Ronen Eldan, Suriya Gunasekar, Michael Harrison, Russell J Hewett, Mojan Javaheripi, Piero Kauffmann, and 1 others. 2024. Phi-4 technical report. *arXiv preprint arXiv:2412.08905*.

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, and 1 others. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Megha Agarwal, Asfandyar Qureshi, Nikhil Sardana, Linden Li, Julian Quevedo, and Daya Khudia. 2023. Llm inference performance engineering: Best practices. Accessed: 2026-01-05.

Rakesh Agrawal, Tomasz Imieliński, and Arun Swami. 1993. Mining association rules between sets of items in large databases. In *Proceedings of the 1993 ACM SIGMOD international conference on Management of data*, pages 207–216.

Rakesh Agrawal and Ramakrishnan Srikant. 1995. Mining sequential patterns. In *Proceedings of the eleventh international conference on data engineering*, pages 3–14. IEEE.

Yuyang Bai, Shangbin Feng, Vidhisha Balachandran, Zhaoxuan Tan, Shiqi Lou, Tianxing He, and Yulia Tsvetkov. 2024. Kgquiz: Evaluating the generalization of encoded knowledge in large language models. In *Proceedings of the ACM Web Conference 2024*, pages 2226–2237.

Ram B Basnet, Clayton Johnson, and Tenzin Doleck. 2022. Dropout prediction in moocs using deep learning and machine learning. *Education and Information Technologies*, 27(8):11499–11513.

Denny Britz, Anna Goldie, Minh-Thang Luong, and Quoc Le. 2017. Massive exploration of neural machine translation architectures. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Stuart K Card. 2018. *The psychology of human-computer interaction*. Crc Press.

Ben Cottier, Ben Snodin, David Owen, and Tom Adamczewski. 2025. Llm inference prices have fallen rapidly but unequally across tasks. Accessed: 2026-01-05.

Boris Cuke, Bart Goethals, and Céline Robardet. 2009. A new constraint for mining sets in sequences. In *Proceedings of the 2009 SIAM international conference on data mining*, pages 317–328. SIAM.

Xiang Deng, Yu Gu, Boyuan Zheng, Shijie Chen, Sam Stevens, Boshi Wang, Huan Sun, and Yu Su. 2023. Mind2web: Towards a generalist agent for the web. *Advances in Neural Information Processing Systems*, 36:28091–28114.

Himel Dev and Zhicheng Liu. 2017. Identifying frequent user tasks from application logs. In *Proceedings of the 22nd international conference on intelligent user interfaces*, pages 263–273.

Wenzheng Feng, Jie Tang, Tracy Xiao Liu, Shuhuai Zhang, and Jian Guan. 2019. Understanding dropouts in moocs. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence*.

Shai Fine, Yoram Singer, and Naftali Tishby. 1998. The hierarchical hidden markov model: Analysis and applications. *Machine learning*, 32(1):41–62.

Qian Fu, Zhanghao Gao, Junyi Zhou, and Yafeng Zheng. 2021. Clsa: A novel deep learning model for mooc dropout prediction. *Computers & Electrical Engineering*, 94:107315.

Haixuan Guo, Shuhan Yuan, and Xintao Wu. 2021. Logbert: Log anomaly detection via bert. In *2021 international joint conference on neural networks (IJCNN)*, pages 1–8. IEEE.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Yuhe Ji, Yilun Liu, Feiyu Yao, Minggui He, Shimin Tao, Xiaofeng Zhao, Chang Su, Xinhua Yang, Weibin Meng, Yuming Xie, and 1 others. 2025. Adapting large language models to log analysis with interpretable domain knowledge. In *Proceedings of the 34th ACM International Conference on Information and Knowledge Management*, pages 1135–1144.

Ming Jin, Shiyu Wang, Lintao Ma, Zhixuan Chu, James Y Zhang, Xiaoming Shi, Pin-Yu Chen, Yuxuan Liang, Yuan-Fang Li, Shirui Pan, and Qingsong Wen. 2024. Time-LLM: Time series forecasting by reprogramming large language models. In *International Conference on Learning Representations (ICLR)*.

Rachneet Kaur, Zhen Zeng, Tucker Balch, and Manuela Veloso. 2025. Lets-c: Leveraging text embedding for time series classification. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 32365–32399.

Kundan Krishna, Deepali Jain, Sanket V Mehta, and Sunav Choudhary. 2018. An lstm based system for prediction of human activities with durations. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 1(4):1–31.

Van-Hoang Le and Hongyu Zhang. 2023. Log parsing with prompt-based few-shot learning. In *2023 IEEE/ACM 45th International Conference on Software Engineering (ICSE)*, pages 2438–2449. IEEE.

Ziqian Lin and Kangwook Lee. 2024. Dual operating modes of in-context learning. In *Forty-first International Conference on Machine Learning*.

Haoxin Liu, Zhiyuan Zhao, Jindong Wang, Harshavardhan Kamarthi, and B Aditya Prakash. 2024a. Lst-prompt: Large language models as zero-shot time series forecasters by long-short-term prompting. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 7832–7840.

Nelson F Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2024b. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173.

Heikki Mannila, Hannu Toivonen, and A Inkeri Verkamo. 1997. Discovery of frequent episodes in event sequences. *Data mining and knowledge discovery*, 1(3):259–289.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Sewon Min, Xinxi Lyu, Ari Holtzman, Mikel Artetxe, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2022. Rethinking the role of demonstrations: What makes in-context learning work? *arXiv preprint arXiv:2202.12837*.

Seungwhan Moon, Andrea Madotto, Zhaojiang Lin, Tushar Nagarajan, Matt Smith, Shashank Jain, Chun-Fu Yeh, Prakash Murugesan, Peyman Heidari, Yue Liu, and 1 others. 2024. Anymal: An efficient and scalable any-modality augmented language model. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pages 1314–1332.

Aadhavan M Nambhi, Bhanu Prakash Reddy, Aarsh Prakash Agarwal, Gaurav Verma, Harvineet Singh, and Iftikhar Ahamath Burhanuddin. 2019. Stuck? no worries! task-aware command recommendation and proactive help for analysts. In *Proceedings of the 27th ACM Conference on User Modeling, Adaptation and Personalization*, pages 271–275.

OpenAI. 2022. New and improved embedding model. Accessed: 2026-01-05.

OpenAI. 2024. Hello gpt-4o. Accessed: 2026-01-05.

OpenAI. 2025. gpt-oss-120b gpt-oss-20b model card. *Preprint*, arXiv:2508.10925.

Sascha Rothe, Shashi Narayan, and Aliaksei Severyn. 2020. Leveraging pre-trained checkpoints for sequence generation tasks. *Transactions of the Association for Computational Linguistics*, 8:264–280.

Jetze Schuurmans and Flavius Frasincar. 2019. Intent classification for dialogue utterances. *IEEE Intelligent Systems*, 35(1):82–88.

Iulian Vlad Serban, Ryan Lowe, Peter Henderson, Laurent Charlin, and Joelle Pineau. 2015. A survey of available corpora for building data-driven dialogue systems. *arXiv preprint arXiv:1512.05742*.

Chirag Shah, Ryen White, Reid Andersen, Georg Buscher, Scott Counts, Sarkar Das, Ali Montazer, Sathish Manivannan, Jennifer Neville, Nagu Rangan, and 1 others. 2025. Using large language models to generate, validate, and apply user intent taxonomies. *ACM Transactions on the Web*, 19(3):1–29.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. *Advances in neural information processing systems*, 27.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Vellum. 2025. Open llm leaderboard. https://www.vellum.ai/open-llm-leaderboard. Accessed: 2026-01-05.

Gaurav Verma, Minje Choi, Kartik Sharma, Jamelle Watson-Daniels, Sejoon Oh, and Srijan Kumar. 2024. Cross-modal projection in multimodal llms doesn't really project visual attributes to textual space. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 657–664.

Mengting Wan, Tara Safavi, Sujay Kumar Jauhar, Yujin Kim, Scott Counts, Jennifer Neville, Siddharth Suri, Chirag Shah, Ryen W White, Longqi Yang, and 1 others. 2024. Tnt-llm: Text mining at scale with large language models. In *Proceedings of the 30th ACM SIGKDD conference on knowledge discovery and data mining*, pages 5836–5847.

Yaqing Wang, Song Wang, Yanyan Li, and Dejing Dou. 2022. Recognizing medical search query intent by few-shot learning. In *Proceedings of the 45th International ACM SIGIR Conference on research and development in information Retrieval*, pages 502–512.

Yaqing Wang, Quanming Yao, James T Kwok, and Lionel M Ni. 2020. Generalizing from a few examples: A survey on few-shot learning. *ACM computing surveys (csur)*, 53(3):1–34.

Jiong Yang, Wei Wang, Philip S Yu, and Jiawei Han. 2002. Mining long sequential patterns in a noisy environment. In *Proceedings of the 2002 ACM SIGMOD international conference on Management of data*, pages 406–417.

Shu Yang, YinFeng Xiao, and Fei Meng. 2024. Deep learning-based method for predicting student dropouts in moocs. In *2024 7th International Conference on Machine Learning and Natural Language Processing (MLNLP)*, pages 1–6. IEEE.

Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc V Le, and 1 others. Least-to-most prompting enables complex reasoning in large language models. In *The Eleventh International Conference on Learning Representations*.

Yihan Zhou, Yan Chen, Xuanming Rao, Yukang Zhou, Yuxin Li, and Chao Hu. 2024. Leveraging large language models and bert for log parsing and anomaly detection. *Mathematics*.

Yichen Zhu, Weibin Meng, Ying Liu, Shenglin Zhang, Tao Han, Shimin Tao, and Dan Pei. 2021. Unilog: Deploy one model and specialize it for all log analysis tasks. *arXiv preprint arXiv:2112.03159*.

A Appendix

A.1 Comparison against fine-tuned baselines

While the core value proposition of WorkflowView is its zero-shot, cross-domain applicability, for completeness, we benchmark the performance against domain-specific fine-tuned baselines. For the browser task inference dataset, we compare against LSTM and BERT-based variants of sequence-to-sequence models (Sutskever et al., 2014). For the MOOC dropout prediction, we compare against several approaches proposed in prior work that use different feature sets to perform the binary classification task, while keeping the evaluation settings consistent.

A.1.1 Baselines for browser task inference

Since generating the task description using browser action sequences is a sequence-to-sequence task (akin to neural machine translation), we use existing the seq2seq models and their implementations. Using the train set of the Mind2Web dataset, the models learn the transformation of action sequences to the task description (word by word), such that actions are demarcated using the `[ACTION]` token. We use word2vec (Mikolov et al., 2013) embeddings, while randomly initializing the out-of-vocabulary words and keeping the embeddings trainable (as some of the activity log vocabulary is not aligned with conventional language vocabulary). To avoid extensive hyperparameter tuning, we follow the training recipe and best practices described by Britz et al. (2017) for LSTM-seq2seq and Rothe et al. (2020) for BERT-seq2seq closely; code available at https://github.com/google/seq2seq and https://github.com/google-research/google-research/tree/master/bertseq2seq,

Models	Metrics	
	MRR	Recall@1
LSTM seq2seq	0.54 ($\pm$0.19)	0.49 ($\pm$0.21)
BERT seq2seq	0.68 ($\pm$0.16)	0.65 ($\pm$0.18)
WorkflowView	0.90 ($\pm$0.08)	0.86 ($\pm$0.13)

Table 4: Comparing our training-free, zero-shot approach (WorkflowView) with domain-specific fine-tuned baselines for task inference using browser action sequences.

respectively. Our evaluations (shown in Table 4) are centered around the same metrics for the 'Global' setting that is described in Section 4.1.

A.1.2 Baselines for MOOC dropout prediction

We compare against the Context-aware Feature Interaction Network (CFIN) introduced by Feng et al. (2019). CFIN models each enrollment using two feature groups: (i) learning-activity features $X(u, c)$ extracted from historical logs (primarily statistics over student actions), and (ii) context features $Z(u, c)$ capturing user and course attributes (e.g., demographics and course category). CFIN combines context-aware smoothing and feature-interaction modeling, and uses a *3-layer* deep neural network (DNN) classifier as its prediction head. As their strongest variant, Feng et al. propose an ensemble strategy ("CFIN-en") analogous to stacking: they take the representation from the penultimate DNN layer and train an XGBoost classifier jointly on this representation and the original features (X, Z). We use the authors' public implementation `https://github.com/wzfhaha/dropout_prediction` and re-train CFIN and the XGBoost-stacked ensemble using features constructed under our 6-day input/24-hour evaluation window, while otherwise following the paper/code defaults. We additionally compare against recent deep learning baselines that operate on week-level temporal feature vectors. Following Yang et al. (2024) (CNN-LSTM) and its bi-attention variant (CNN-LSTM Bi-Att). We keep the architecture and hyperparameters consistent with the original work (e.g., dropout and early stopping), and reconstruct the inputs to match our 6-day input/24-hour evaluation window. Table A.1.2 shows the comparison; WorkflowView achieves its best performance with 5 in-context examples per class (10 total), reaching a weighted $\mathcal{F}_1$ of 0.90. It is noteworthy that the performance is competitive even at 0-shot and remains stable across a range of fewshot budgets (see Fig. 2 & 3).

Models	Weighted F_1
DNN (3-layer MLP) (Feng et al., 2019)	0.83
CFIN-en (Feng et al., 2019)	0.90
CNN-LSTM (Yang et al., 2024)	0.86
CNN-LSTM Bi-Att (Yang et al., 2024)	0.87
WorkflowView (0-shot)	0.84
WorkflowView (5-shot)	0.90

Table 5: Comparing our few-shot adapted approach to domain-specific supervised baselines for MOOC dropout prediction. Reported values are $\mu(\pm\sigma)$.

A.1.3 Experiments with smaller LLMs

Metric	Phi-4 (14b)	GPT-OSS-20b
MRR	0.89 ($\pm$0.09)	0.90 ($\pm$0.08)
Recall@1	0.85 ($\pm$0.12)	0.86 ($\pm$0.13)
Recall@3	0.93 ($\pm$0.05)	0.94 ($\pm$0.07)
Recall@5	0.95 ($\pm$0.06)	0.96 ($\pm$0.05)
Recall@10	0.97 ($\pm$0.02)	0.98 ($\pm$0.01)

Table 6: Embedding-based retrieval of ground-truth task descriptions using task descriptions generated using WorkflowView. Phi-4 and `gpt-oss-20b` were used to generate the task descriptions.

We evaluate the performance of smaller LLMs on the browser activity inference task ('Global' setting; see Section 4.1) and consider two models — Phi-4 (14B parameters) (Abdin et al., 2024) and `gpt-oss-20b` (OpenAI, 2025). This is largely to assess whether the LLMs that are among the fastest (as measured by tokens per second) and cheapest (cost per token) (Vellum, 2025), can also interpret user activity sequences as well as more expensive counterparts like GPT-4o. We find that the task descriptions generated using Phi-4 and `gpt-oss-20b` demonstrated a mean similarity of 0.902 ± 0.036 and 0.909 ± 0.039, respectively. Table 6 shows that the retrieval-based metrics are also on par with those obtained using the GPT-4o model in Table 1.

A.2 Additional related work

Hierarchical action modeling: Beyond the pattern mining techniques discussed in Section 2, our work shares conceptual roots with hierarchical representation learning for sequential data. Traditional approaches in robotics and plan recognition have long used Hierarchical Hidden Markov Models (HHMMs) (Fine et al., 1998) to decompose complex behaviors. In the human-computer interaction community, GOMS modeling (Card, 2018) provided early foundations for decomposing user goals into operators, which WorkflowView's context involves

replacing manual task analysis with LLM-based inference.

LLMs for system logs: While we discuss LLMs for user activity sequence, there is a growing body of work specifically targeting system logs for anomaly detection and root cause analysis. Methods like LogPPT (Le and Zhang, 2023) and LogLLM (Ji et al., 2025) demonstrate the utility of prompt-based learning for structured log parsing. WorkflowView differs by focusing on semantic user intent across diverse UI domains rather than system-level health monitoring.

In-context learning for sequence tasks and hierarchical prompting: The sensitivity analysis in Section 4.2 regarding few-shot examples aligns with recent findings on the "lost in the middle" phenomenon and context window saturation (Liu et al., 2024b). More recently, the "least-to-most" prompting paradigm (Zhou et al.) demonstrated that LLMs are significantly more effective when complex problems are decomposed into a series of simpler sub-problems, with the solution to each step facilitating the next. WorkflowView translates this principle to the domain of telemetry by treating raw event logs as the input "complex problem" and using a layered architecture to progressively abstract user intent. The current study does not include a formal ablation on the hierarchical structure itself (e.g., comparing single-pass prompting against the multi-layered approach). We maintain that this hierarchical design is central to the "progressive denoising" required for noisy and granular telemetry. As least-to-most prompting has already been shown to significantly outperform single-pass reasoning in tasks requiring complex decomposition and easy-to-hard generalization, we chose to implement this proven hierarchical logic rather than re-evaluating its effectiveness ablations.

A.3 Quality & stability of inferred activities

It is important to address the stability of our unsupervised categorization, particularly within the proprietary case study involving enterprise document software. Because this domain lacks a labeled ground-truth for inferred high-level categories, we conducted a sensitivity analysis to evaluate the consistency of the LLM's discovered categories. We re-ran the entire categorization pipeline across three independent trials using different random seeds. Qualitative analysis of the inferred categories indicates that the "top-N" discovered categories, accounting for over 90% of the total analyzed sequences, were inferred with remarkable consistency across all runs. For instance, core categories such as "Reviewing comments (collaboration)" and "Content transfer within or between documents" appeared in every trial. The variance in these high-frequency categories was limited strictly to lexical changes in naming; for example, one trial labeled a cluster as "Active editing of content" while another named it "Editing content actively," yet both mapped to the same underlying distribution of low-level user action patterns as verified by tf-idf scores of the action sequences associated with these categories. In contrast, the "long-tail" activities representing less than 10% of the dataset exhibited higher instability; for example, "Document navigation" was not consistently isolated as a standalone category across all runs, often being absorbed into broader clusters. While future work could involve extensive human evaluation to further validate these semantic boundaries, the qualitative results observed across our experiments—particularly the browser inference tasks in Table 2 and Table 11—are highly compelling. The precision with which the framework translates low-level user activity sequences into human-readable intent builds significant confidence in the stability and quality of these inferred categories for functional product telemetry, where the goal is to understand the most common user journeys.

Prompt Type	Content
System Prompt	You are an expert at analyzing web interaction patterns and translating technical UI action logs into clear, human-readable descriptions of user behavior and intent.
User Prompt	You are analyzing web interaction telemetry data from user sessions. Your task is to provide a detailed natural language description of what the user is doing on a website based on the sequence of UI actions they performed. **Action sequence (in order):** {action_text} Each action follows the format: [element_type] element_description → ACTION_TYPE: optional_value **Instructions:** • Provide a detailed, step-by-step description of what the user is doing • Focus on the user's workflow and intent behind the actions • Interpret technical UI elements into plain language • Pay attention to the sequence and progression of actions • Describe what the user is trying to accomplish through these interactions • Include details about form filling, navigation, searches, selections, etc. Provide a comprehensive description of the user's web interaction workflow.

Table 7: WorkflowView Layer 1 prompt for browser task inference; obtaining natural language descriptions from action sequences.

Prompt Type	Content
System Prompt	You are an expert at distilling detailed user workflow descriptions into concise, natural task descriptions that capture the user's primary intent.
User Prompt	Based on the detailed workflow analysis below, generate a concise task description that captures what the user is trying to accomplish. The task description should be similar to how a user would naturally describe their goal when using a website. **Detailed workflow analysis:** {detailed_description} **Instructions:** • Generate ONLY the task description, nothing else • Make it concise and action-oriented • Focus on the end goal, not the individual steps • Use natural language that describes the user's intent • Do not mention the website name, domain, or technical details • Format it as a simple sentence or phrase describing the task • Include the necessary details that are required to successfully complete the task **Task description:**

Table 8: WorkflowView Layer 2 Prompt for browser task inference; obtaining succinct summary of user intent.

Prompt Type	Content
System Prompt	You are an expert at analyzing online learning behavior patterns and translating technical activity logs into clear, human-readable descriptions of student engagement and learning patterns.
User Prompt	You are analyzing MOOC (Massive Open Online Course) learning behavior data. Your task is to provide a detailed natural language description of a student's learning activities and engagement patterns based on their chronological sequence of actions. **Time Window Analyzed:** {hours_start_before_last} to {hours_end_before_last} hours before last action **TARGET STUDENT'S ACTIVITY SEQUENCE:** {action_text} **Instructions:** • Provide a detailed, step-by-step description of what the student is doing • Focus on the student's workflow and intent behind the actions • Interpret technical UI elements into plain language • Consider the context of students interacting with MOOC • Pay attention to the sequence and progression of actions, their frequency, and timestamps to note regularity / sparsity / consistency / inconsistency Provide a comprehensive and balanced description of the student's MOOC learning behavior and engagement patterns.

Table 9: WorkflowView Layer for MOOC student dropout prediction; obtaining natural language descriptions.

Prompt Type	Content
System Prompt	You are an expert at summarizing student learning behavior patterns into concise, actionable insights about student engagement in online courses.
User Prompt	Based on the detailed learning behavior analysis below, generate a concise summary that captures the key patterns of student engagement and learning behavior in this MOOC course. **Time Window:** {hours_start_before_last} to {hours_end_before_last} hours before last action **TARGET STUDENT'S DETAILED ANALYSIS:** {detailed_description} **Instructions:** • Generate a concise summary (2–3 sentences) of the student's engagement patterns • Focus on key behavioral indicators and learning activity patterns • Highlight temporal patterns and engagement levels • Concisely surface insights that may help infer if the student will dropout of the course of continue based on their engagement so far **Engagement Summary:**

Table 10: WorkflowView Layer 2 for MOOC student dropout prediction; obtaining succinct activity summary.

1. Action Sequence	[svg] → CLICK, [link] Your lists → CLICK, [link] Create a list → CLICK, [svg] → CLICK, [span] Walgreens → CLICK, [textbox] Add a title (Required) → TYPE: Walgreens, [img] → CLICK, [button] Next → CLICK, [link] Personal Care → CLICK, [svg] → CLICK, [img] → CLICK, [span] Add to list → CLICK, [checkbox] Walgreens New → CLICK, [button] Done → CLICK, [path] → CLICK, [path] → CLICK, [path] → CLICK, [svg] → CLICK, [img] → CLICK, [span] Add to list → CLICK, [checkbox] Walgreens New → CLICK, [button] Done → CLICK, [path] → CLICK, [link] View More → CLICK, [img] → CLICK, [span] Add to list → CLICK, [checkbox] Walgreens New → CLICK, [button] Done → CLICK, [button] Back → CLICK, [path] → CLICK, [link] Shower Essentials → CLICK, [img] → CLICK, [span] Add to list → CLICK, [checkbox] Walgreens New → CLICK, [button] Done → CLICK, [button] Back → CLICK, [link] Lists → CLICK
Generated Task Description	Create a Walgreens shopping list and add personal care and shower essentials items.
Ground-truth Task Description	Create a new list and add four items from the personal care category at Walgreens.
2. Action Sequence	[link] SEARCH CARS → CLICK, [button] Distance & Shipping → CLICK, [button] Change Location → CLICK, [textbox] Enter ZIP or State → CLICK, [textbox] Enter ZIP or State → TYPE: 07470, [button] set store → CLICK, [button] Distance & Shipping → CLICK, [button] Nationwide → CLICK, [menuitem] 100 miles → CLICK, [input] → CLICK, [button] Back to all categories → CLICK, [button] Make → CLICK, [listitem] Honda → CLICK, [button] Back to all categories → CLICK, [button] Model → CLICK, [listitem] Civic → CLICK, [button] Back to all categories → CLICK, [button] Year → CLICK, [button] 2010 → CLICK, [menuitem] 2017 → CLICK, [button] 2023 → CLICK, [menuitem] 2017 → CLICK, [button] Back to all categories → CLICK, [button] Features → CLICK, [listitem] Sunroof(s) → CLICK, [button] Back to all categories → CLICK, [button] Exterior Color → CLICK, [listitem] Black → CLICK, [button] Back to all categories → CLICK, [button] Sort by → CLICK, [label] Lowest price → CLICK
Generated Task Description	Find a black 2017 Honda Civic with a sunroof within 100 miles of Wayne, NJ, sorted by lowest price.
Ground-truth Task Description	Find the cheapest 2017 Honda Civic within 100 miles of 07470 which has a sun roof and black exterior color while excluding out of market vehicles
3. Action Sequence	[textbox] DESTINATION → TYPE: NIAGRA FALLS, [option] Niagara Falls, ON, Canada → CLICK, [button] Date selection → CLICK, [span] 17 → CLICK, [span] 20 → CLICK, [button] Done → CLICK, [button] 1 Room, 1 Guest → CLICK, [button] Add Rooms → CLICK, [button] Add Adults → CLICK, [button] Add Adults → CLICK, [button] Add Children → CLICK, [button] Add Age → CLICK, [button] Add Less than 1 → CLICK, [button] Add 1 → CLICK, [button] Add 2 → CLICK, [button] Find Hotels → CLICK, [checkbox] Show rates with taxes and all fees → CLICK, [checkbox] Show available hotels only → CLICK, [link] Free breakfast → CLICK, [link] Pool → CLICK, [generic] Price → CLICK, [label] 100 - 200 USD → CLICK, [button] APPLY → CLICK, [combobox] Sort by → CLICK, [option] Price → CLICK, [link] VIEW RATES → CLICK, [button] Member Rate Prepay Non-refundable → CLICK, [label] Accept cancellation → CLICK, [button] CONTINUE → CLICK
Generated Task Description	Book a hotel in Niagara Falls, ON, for three adults and three children from April 17 to April 20 with free breakfast and a pool, within a $100-$200 budget after including taxes and fees.
Ground-truth Task Description	Find two rooms in a cheapest hotel in Niagra Falls for three adults and one three year old kid from May 17 to May 20, view only available hotels within 100 to 200 dollar range with taxes and fees, and choose the cheapest hotel that offers free breakfast and a pool.
4. Action Sequence	[link] Shop → CLICK, [img] Sports car icon → CLICK, [button] Sort by → CLICK, [label] Lowest price → CLICK, [button] Back to all categories → CLICK, [button] Fuel Type → CLICK, [listitem] Gas → CLICK, [button] Back to all categories → CLICK, [button] Year → CLICK, [button] 2010 → CLICK, [menuitem] 2018 → CLICK, [button] 2023 → CLICK, [menuitem] 2022 → CLICK, [button] Back to all categories → CLICK, [button] Exterior Color → CLICK, [listitem] Gray → CLICK, [button] Back to all categories → CLICK, [button] Transmission → CLICK, [span] Automatic → CLICK, [button] Back to all categories → CLICK, [heading] Distance & Shipping → CLICK, [button] $99 Or Less → CLICK, [menuitem] Free to home or store → CLICK, [button] Back to all categories → CLICK, [switch] COMPARE → CLICK, [path] → CLICK, [button] Add to Compare → CLICK, [button] Go to Compare → CLICK, [button] COMPARE PHOTOS → CLICK
Generated Task Description	Find and compare affordable gray automatic sports cars that run on gas from recent years (2018-2022) with no shipping costs.
Ground-truth Task Description	Search for an automatic grey sports car with the lowest price, gas fuel and free shipping manufactured between 2018 to 2022, compare the top two results and compare photos.
5. Action Sequence	[button] hotels → CLICK, [div] Destination or property → TYPE: jakarta, [hp-input-button] Destination or property → TYPE: jakarta, [div] Jakarta → CLICK, [div] Choose date → CLICK, [div] Jun → CLICK, [generic] 1 → CLICK, [generic] 4 → CLICK, [button] Search → CLICK, [button] Yes, I agree → CLICK, [span] Lowest price → CLICK, [button] Choose room → CLICK, [button] Book now → CLICK, [textbox] First name → TYPE: Joe, [textbox] Surname → TYPE: Bloggs, [textbox] Email address → TYPE: buckeye.foobar@gmail.com, [textbox] Confirm email address → TYPE: buckeye.foobar@gmail.com, [input] → TYPE: 1111111111111111, [textbox] Address 1 → TYPE: the home of joe bloggs, [textbox] City → TYPE: new york, [textbox] Postcode/ZIP code → TYPE: 10001, [combobox] State → TYPE: new york, [textbox] Card number → TYPE: 1234, [combobox] Card type → SELECT: MasterCard, [combobox] Month → SELECT: 01, [combobox] Year → SELECT: 2023, [textbox] CVV → TYPE: 123, [textbox] Cardholder's name → TYPE: joe bloggs, [svg] → CLICK
Generated Task Description	Book a hotel in Jakarta from June 1 to June 4 at the lowest price, for Joe Bloggs with email buckeye.foobar@gmail.com and phone number 11111111111. The billing address is specified to be in New York, 10001.
Ground-truth Task Description	Book the cheapest available hotel for a three night stay from 1st June in Jakarta. The guest is named Joe Bloggs with the email address of buckeye.foobar@gmail.com and phone number of 11111111111. Billing address is in New York, zip code 10001.

Table 11: Qualitative examples of task descriptions generated using WorkflowView from the action sequences, and the corresponding ground-truth task descriptions.