Learning Structural Regularities over Mobile UI Flows from Interaction Traces

Iskandar Salama, Masayasu Atsumi
Soka University, Tokyo, Japan

***Abstract*—Mobile applications exhibit rich user interface (UI) flows composed of sequences of screens connected through user interactions. While prior work has made significant progress in understanding individual UI screens, reusable flow-level regularities across applications remain underexplored. In this study, we learn a role-based probabilistic prior over mobile UI flows from large-scale interaction traces by mapping each screen to an abstract screen role via unsupervised clustering of multimodal screen embeddings derived from screenshots and view-hierarchy text. Interaction traces are then converted into sequences of screen roles, enabling probabilistic modeling of flow structure beyond app-specific identifiers and layouts. We study two complementary tasks. First, for next-step prediction on unseen applications and held-out categories, simple n-gram baselines already capture meaningful cross-app regularities, and a causal Transformer further improves performance, achieving R@1 = 0.2598 and R@10 = 0.7268 on unseen test applications at K = 40 while outperforming the trigram baseline across all reported cutoffs. Second, we study atypical-transition scoring under a fixed sampled-candidate protocol, where the same learned flow prior is used to rank observed transitions against sampled alternatives. Under this controlled setting, the Transformer achieves the strongest ranking performance among the compared models. These results indicate that role-based abstractions provide a promising basis for modeling UI flow regularities across applications and for ranking-oriented sequence prediction under cross-application generalization. The anomaly results are encouraging under the sampled evaluation protocol, but broader validation will require more realistic evaluation settings and structured human assessment.**

Keywords*—UI Flow modeling; causal transformer; n-grams; clustering; multimodal embeddings; next-step prediction; cross-app generalization

I. Introduction

Modern mobile applications exhibit rich and complex user interface (UI) flows composed of sequences of screens connected through user interactions. While individual screens can often be interpreted in isolation, their functional meaning and usability are strongly shaped by their position within a larger flow. For example, a visually similar screen may serve as onboarding, authentication, or confirmation depending on what precedes and follows it. Understanding such flow-level structure is therefore critical for tasks ranging from design inspection and usability evaluation to automated testing and design assistance.

Recent years have seen substantial progress in large-scale UI analysis enabled by datasets that combine screenshots, view hierarchies, and interaction traces. A key catalyst is the RICO dataset, which provides such multimodal UI data across thousands of Android applications [1]. These data have motivated advances in screen representation learning and semantic understanding of user interfaces, and make it possible to study UI structure at the level of screen composition and navigation rather than individual screens alone [2], [3].

However, the majority of prior work emphasizes *screen-level* modeling: learning screen embeddings, categories, or textual summaries for individual screens based on visual appearance, textual content, or view hierarchy structure [1], [4], [3]. In contrast, *app-level flow structure*—how screens are arranged and traversed as part of user journeys—remains comparatively under-modeled as a reusable prior across applications.

A central challenge is generalization across applications. UI flows vary widely in concrete implementation, yet many apps share abstract structural regularities (e.g., onboarding followed by authentication, transitions from search results to a detail view, and destructive actions requiring confirmation). Capturing such regularities requires abstracting away app-specific identifiers while retaining functional roles and temporal dependencies. Existing work either models navigation within a single app (e.g., via storyboards or static graphs) [5], predicts low-level user actions (e.g., next click) [6], or targets adjacent objectives such as UI similarity/transition event understanding [7], limiting its suitability as a transferable flow prior.

In this study, we learn a *structural prior over UI flows*—that is, an empirically learned distribution over likely flow continuations—from large-scale interaction traces by operating at the level of abstract *screen roles*. Each screen is mapped to a discrete role obtained via unsupervised clustering of multimodal screen embeddings derived from screenshots and screen text (e.g., view-hierarchy text). Interaction traces are then converted into sequences of role identifiers, enabling the learning of probabilistic flow models that operate over a reduced state space shared across applications.

Using this representation, we study two complementary problems. First, we examine next-step prediction: given a partial UI flow, can a model trained on other apps correctly rank plausible next screen roles? Second, we investigate atypical transition mining by scoring observed transitions under the learned prior and identifying structurally unusual steps that may indicate missing screens, design inconsistencies, or legitimate but rare flows. We explore both classical n-gram baselines and Transformer-based sequence models, and we evaluate cross-app generalization under app-level splits.

Our contributions are threefold:

- A multimodal clustering pipeline that maps heterogeneous UI screens into reusable screen roles suitable for flow modeling.
- Evidence that statistical and neural sequence models

trained on role sequences capture meaningful cross-app UI flow regularities, improving next-step suggestion beyond short-context baselines.

- An anomaly scoring formulation that uses the learned flow prior to mine atypical UI transitions for design inspection.

II. Related Work

A. Screen Representation and Clustering

Large-scale analysis of mobile user interfaces became feasible through advances in automated interaction mining and dataset construction. ERICA introduced a scalable system for crawling mobile applications and mining interaction traces without app instrumentation [8]. Building on this line of work, the RICO dataset standardized a large corpus of mobile UI screenshots, view hierarchies, and interaction traces across thousands of Android applications [1]. RICO has since served as a foundation for research in screen representation, UI similarity, and design analysis.

Other work explores multimodal approaches to screen understanding and language grounding. Screen2Words formulates mobile UI summarization, generating concise natural-language descriptions from multimodal UI inputs (image, text, and structure) [3]. While such summaries provide interpretability, they are not designed for flow modeling and introduce additional ambiguity due to linguistic variability.

More closely related are approaches that use interaction traces to shape screen representations. Screen2Vec learns semantic embeddings for GUI screens and components in a self-supervised manner using interaction-trace context, enabling cross-app similarity and compositional representations [2]. While these representations implicitly encode contextual regularities, they do not explicitly define a probabilistic flow model for next-step prediction or atypical transition scoring. In contrast, our work uses multimodal embeddings as an intermediate representation and discretizes them into screen roles via clustering to enable explicit sequence modeling over role tokens.

B. Modeling UI Behavior from Interaction Traces

Interaction traces have been widely used to study user behavior and optimize UIs. Several works formulate next-action or next-click prediction, learning to predict the next UI element a user will interact with based on click sequences and screen context [6]. Such methods typically operate at the granularity of widgets or coordinates and are often tailored to behavioral modeling or personalization.

Other efforts analyze traces to extract task flows or navigation graphs to support automated testing and program understanding. For example, AppFlow synthesizes reusable UI tests by recognizing common screens and widgets across applications, leveraging learned abstractions to improve robustness [9]. Separately, StoryDroid uses static analysis to extract an app's activity transition graph and generate storyboard-style summaries [5]. These systems are valuable for testing and comprehension, but they primarily target per-app structure extraction or task-specific reuse rather than learning a reusable probabilistic prior over generic UI flow structure.

Work on understanding screen relationships also intersects with flow modeling. Feiz et al. study screen similarity and transition event understanding from screenshots, achieving strong performance on identifying screen relationships and transition events [7]. This line of work improves trace understanding but does not model multi-step flow continuations as ranked next-role candidates. Our approach instead focuses on learning cross-app regularities as a next-step distribution over abstract roles conditioned on longer history.

C. Design Assessment and UI Automation

A related line of work leverages pretrained vision–language models for UI design assessment. For example, UIClip proposes a data-driven model to score UI design quality and relevance given a screenshot and a natural language description [10]. These systems primarily focus on static properties (quality, aesthetics, relevance) rather than navigational structure.

A related but orthogonal direction studies within-screen structural completion rather than flow-level navigation. For example, LayCoder formulates mobile UI layout completion as masked structured prediction with an encoder-only Transformer and a layout tokenizer, targeting element-level completion within a single screen rather than cross-screen flow modeling [11].

Related systems such as Kite extract task models from app demonstrations to build conversational bots, deriving structured task graphs from interaction traces [12]. While conceptually related in using traces to infer higher-level structure, our goal differs: we learn a reusable probabilistic prior over flow continuations and atypical transitions across many apps rather than generating bot-specific task graphs.

D. GUI Agents and Task-Oriented UI Interaction

Recent work has explored treating graphical user interfaces as environments for embodied or language-driven agents. Systems such as SIFT [13] combine vision–language models with action prediction to enable agents to perform multi-step tasks in mobile and desktop applications. These approaches typically train on large collections of screenshots and interaction trajectories, often derived from datasets such as RICO, and evaluate success based on task completion or action accuracy.

While these systems demonstrate impressive generalization in executing unseen tasks, their primary objective is action synthesis rather than structural modeling. They operate directly on pixel-level observations and action spaces, without explicitly learning or exposing a reusable representation of UI flow structure. In contrast, our work focuses on learning an explicit, interpretable structural prior over screen roles and transitions, enabling analysis tasks such as next-step suggestion and atypical transition detection that are not addressed by agent-based formulations.

Overall, our work lies at the intersection of screen representation learning, interaction-trace modeling, and UI analysis. It differs from prior work in three key respects: (1) we explicitly abstract heterogeneous screens into reusable screen roles, (2) we learn a probabilistic model of multi-step flow continuation over these shared role sequences across applications, and (3) we use the same learned model for both next-step prediction

and atypical transition scoring. To our knowledge, this combination has not been directly addressed in prior mobile UI research.

III. Problem Setup and Tasks

We study whether domain-independent regularities exist in mobile UI navigation flows and whether such regularities can be learned from large-scale interaction traces and reused across previously unseen applications. We assume access to interaction traces recorded from many applications and aim to learn a probabilistic *flow prior* over an abstract discrete state space of *screen roles*. This learned prior supports two tasks: 1) next-step suggestion and 2) atypical transition detection.

A. Interaction Traces

Let $\mathcal{A}$ denote a set of applications. Each application $a \in \mathcal{A}$ contains a set of interaction traces $\mathcal{T}_a$. A trace $\tau \in \mathcal{T}_a$ is an ordered sequence of UI screens,

$$\tau = (s_1, s_2, \ldots, s_L),$$

where, L is the sequence length. Each screen s_t contains (at minimum) a screenshot image I and a structured view hierarchy H (e.g., a UI tree with widget metadata and visible text).

B. Screen Roles as a Discrete Abstraction

Directly modeling flows over raw screens is brittle due to high-dimensional, app-specific variation. We therefore introduce a discrete abstraction layer of *screen roles*, where each observed screen is mapped to one of K role IDs.

Concretely, we assume a mapping from screens to role assignments $c_t \in \{1, 2, \ldots, K\}$. Intuitively, roles are intended to capture reusable UI states (e.g., "list-like browsing", "detail view", "login gate") without requiring manual semantic labels. Given a trace $\tau = (s_1, \ldots, s_L)$, the abstraction yields a corresponding role sequence $\mathbf{c} = (c_1, \ldots, c_L)$.

C. Flow Prior Over Role Sequences

We model UI navigation as a conditional distribution over the next role given the role history:

$$p_\theta(c_{t+1} \mid c_{1:t}), \tag{1}$$

which represents a predictive distribution over the next screen role. We refer to this learned distribution as a *flow prior*, as it captures typical navigation patterns across applications.

D. Task 1: Next-Step Suggestion

Given a role-sequence $c_{1:t}$, the model ranks candidate next roles $j \in \{1, \ldots, K\}$ by $p_\theta(j \mid c_{1:t})$ and returns the top-k roles as *next-step suggestions*. We evaluate this as a retrieval problem using Recall@k: the fraction of ground-truth next roles c_{t+1} that appear in the model's top-k suggestions over all evaluated transitions. Recall@k is well-suited to UI navigation because multiple next roles may be plausible given hidden user intent and system state.

E. Task 2: Atypical Transition Detection

Beyond next-step suggestion, we aim to flag suspicious or atypical transitions that may indicate missing intermediate steps, broken redirects, or rare navigation patterns requiring review. Given an observed transition from context $c_{1:t}$ to a realized next role c_{t+1}, we score atypicality using the model's predicted probability for the next role; transitions with lower probability are treated as more atypical. We evaluate this task using ranking-based metrics described in Section VI.

IV. Screen Representation and Role Clustering

Our approach builds a reusable flow prior by first mapping each raw screen to a discrete *screen role*. This requires 1) a multimodal screen representation that captures visual appearance and UI content/structure, and 2) an unsupervised clustering procedure that discretizes the continuous representation space into K roles.

Fig. 1 illustrates the entire pipeline. In this section, we focus on the first stage of the pipeline, where each screen is embedded using multimodal features and clustered into discrete screen roles. These roles serve as the abstraction used later for sequence modeling of UI flows.

A. Multimodal Screen Representation

A screen contains both visual information (layout, icons, imagery) and structured semantic information (widget types and text). We represent a screen $s = (I, H)$ by fusing an image embedding from the screenshot I with a text embedding derived from a serialization of the view hierarchy H. We deliberately avoid training a UI-specific encoder from scratch and instead rely on strong pretrained representation models.

Given a screenshot I, we compute a visual embedding using the CLIP image encoder [14]. We use the ViT-L/14 variant and extract a 768-dimensional embedding, followed by ℓ_2 normalization.

A UI view hierarchy is provided as a JSON tree describing UI elements (widget types) and their textual properties. We convert this hierarchy into a deterministic textual sequence by traversing the UI tree and extracting, for each node, a widget identifier (e.g., button, text field, text view) and an associated text attribute, such as visible text, content description, hint, or label. Each node is serialized as either `ClassName: text` when textual content is available, or simply `ClassName` otherwise. For example, a screen containing a button labeled "Login" and a text field with placeholder "Email" would be serialized as `UI_SCREEN <SEP> Button: Login <SEP> EditText: Email`, where `<SEP>` is used only for readability; in the actual implementation, entries are separated by newline characters.

The resulting sequence is then embedded using Sentence-BERT (SBERT) [15], specifically all-MiniLM-L6-v2 in our implementation, producing a 384-dimensional embedding that is again ℓ_2 normalized.

Let $v \in \mathbb{R}^{d_v}$ and $t \in \mathbb{R}^{d_t}$ denote the normalized CLIP and SBERT embeddings, respectively. We form a late-fusion representation by concatenation:

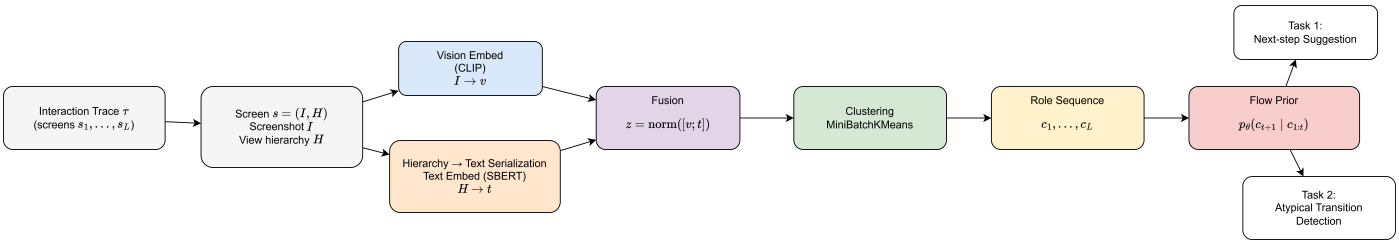

Fig. 1. Overview of the proposed pipeline. Screens are embedded using multimodal representations, clustered into discrete screen roles, converted into role sequences, and modeled with a flow prior to support next-step suggestion and atypical transition detection.

$$z = \mathrm{norm}\big([v; t]\big) \in \mathbb{R}^{d_v + d_t}, \tag{2}$$

where, $\mathrm{norm}(\cdot)$ denotes ℓ_2 normalization. Concatenation preserves modality-specific information while producing a single vector suitable for standard clustering. Intuitively, visual embeddings alone may over-emphasize template similarity, while text/structure embeddings alone may overfit app-specific vocabulary. We therefore use the fused representation as our default operating choice for the main pipeline, and we later assess this design empirically through a modality ablation at the selected operating point $K = 40$.

B. Screen Role Clustering

Given fused embeddings $\{z_i\}_{i=1}^{N}$, we cluster screens into K roles and assign each screen a discrete role ID $c_i \in \{1, \ldots, K\}$. We use MiniBatchKMeans [16], [17] for scalability on large screen corpora. Each screen is assigned to the nearest centroid under Euclidean distance (equivalently cosine distance under ℓ_2 normalization).

The choice of K controls an abstraction trade-off: small K yields coarse roles that improve statistical strength and cross-app generalization, while large K yields fine roles that increase sparsity and can reduce generalization in downstream transition modeling. We therefore treat K as a key hyperparameter and report results across multiple values (e.g., $K \in \{20, 40, 80, 120, 200\}$) in Section VI.

C. Role Cluster Diagnostics

Because screen roles are learned without labels, we assess clustering quality using complementary diagnostics rather than a single criterion. We compare standard internal validation metrics across candidate values of K, including the Silhouette score [18], Davies–Bouldin (DB) index [19], and Calinski–Harabasz (CH) index [20], evaluate stability across random seeds using repeated MiniBatchKMeans runs [16], and inspect cluster-size distributions to detect fragmentation. As summarized in Table I, no single value of K is uniformly best across all internal metrics; instead, the results suggest a trade-off between compactness, stability, and granularity.

These internal metrics are used only as heuristic diagnostics for selecting a practical range of K; they are not treated as direct evidence that the learned roles are semantically valid or optimal for downstream use. In this study, the selected operating point is justified primarily by stability, fragmentation behavior, and downstream predictive utility rather than by any single internal clustering score.

To choose a representative operating point, we prioritize two empirical criteria: 1) stability across repeated clustering runs, and 2) fragmentation behavior as reflected by singleton and very small clusters. We then use downstream next-step prediction results only as a secondary check that the selected role vocabulary remains practically useful. Additional diagnostics at $K = 10$ and $K = 15$ show that smaller values yield stronger compactness-oriented internal metrics, but produce substantially coarser partitions: the median cluster size rises from 2942.5 at $K = 20$ to 4084.8 at $K = 15$ and 6582.5 at $K = 10$, with the largest cluster at $K = 10$ containing $13{,}314$ screens on average across runs. We therefore treat $K = 20$ as the coarse-grained lower bound for downstream flow modeling. Within the practically useful range $K \geq 20$, $K = 40$ provides a substantially finer partition than $K = 20$ while maintaining comparable seed stability (ARI 0.407 vs. 0.380) and higher NMI (0.647 vs. 0.576), and it avoids the stronger fragmentation that appears at larger values of K. Normalized Mutual Information (NMI) [21] is therefore reported alongside ARI as a complementary stability reference.

We therefore use $K = 40$ in the subsequent flow-modeling experiments as the representative operating point balancing role granularity, clustering stability, and downstream predictive utility.

Fig. 2 provides a qualitative view of the fused embedding space: intra-cluster screen pairs tend to have higher cosine similarity than inter-cluster pairs.

To complement these internal diagnostics, we also provide a qualitative audit of the learned roles at $K = 40$ in the supplementary material (Appendix), where each selected cluster is illustrated using centroid-nearest representative screens from the fused embedding space. The examples suggest that several high-frequency clusters correspond to visually and functionally coherent UI states.

V. Flow Modeling with Screen Roles

Using the role assignments obtained in Section IV, each interaction trace is converted into a discrete sequence of role IDs. We then model the flow prior $p_\theta(c_{t+1} \mid c_{1:t})$ using two modeling approaches: 1) count-based n-gram priors and 2) a causal Transformer over role sequences. Both approaches output a next-role distribution that can be used for next-step suggestion and for assigning atypicality scores to observed transitions.

TABLE I. CLUSTER DIAGNOSTICS ACROSS CANDIDATE VALUES OF K

K	Silhouette↑	DB↓	CH↑	ARI↑	NMI↑	Singletons	Median size
10	0.0170	4.5768	1058.43	0.401	0.533	0.0	6582.5
15	0.0145	4.3387	785.32	0.404	0.570	0.0	4084.8
20	0.0136	4.4083	686.97	0.380	0.576	0.0	2942.5
40	0.0030	4.3292	427.74	0.407	0.647	0.0	1599.5
80	0.0077	4.2581	262.30	0.399	0.675	0.4	753.5
120	0.0013	4.0699	196.16	0.355	0.681	0.8	524.0
200	-0.0031	3.9996	134.55	0.347	0.697	1.4	296.5

$K = 10$ and $K = 15$ provide coarser role vocabularies despite stronger compactness-oriented internal metrics, while $K = 40$ serves as the representative operating point within the practically useful range $K \geq 20$. Stability is summarized across 5 MiniBatchKMeans runs with seeds 0–4.

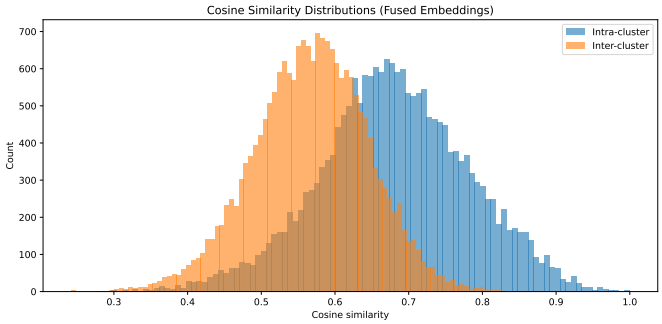

Fig. 2. Intra- vs. inter-cluster cosine similarity distributions for fused screen embeddings (illustrative example with $K = 40$). Intra-cluster pairs are shifted toward higher similarity than inter-cluster pairs, supporting that the learned roles capture consistent patterns rather than arbitrary partitions.

A. Trace-to-Role Sequence Preprocessing

Let $\tau = (s_1, \ldots, s_L)$ be a trace of screens. We map each screen to a role ID via the learned embedding-and-clustering pipeline (Section IV), yielding a role sequence $\mathbf{c} = (c_1, \ldots, c_L)$.

UI traces often contain repeated observations of the same functional state (e.g., re-rendering, minimal changes, or near-duplicate screens). To avoid overweighting self-transitions and to emphasize meaningful navigation steps, consecutive occurrences of the same role are merged. For example, a role sequence such as $(c_1, c_2, c_2, \ldots, c_2, c_3, \ldots)$ is reduced to $(\tilde{c}_1, \tilde{c}_2, \tilde{c}_3, \ldots)$, where $\tilde{c}_i \neq \tilde{c}_{i+1}$. Any trace with fewer than two roles after consecutive-duplicate merging is removed. (Additional dataset filtering and split protocol are described in Section VI.)

B. n-gram Flow Priors

We begin with count-based probabilistic priors over role sequences. These priors are strong, interpretable baselines and provide reasonable next-role probabilities for typicality scoring when the role vocabulary is moderate.

Let $N(\cdot)$ denote counts computed from training role sequences. The *unigram* prior ignores context entirely and ranks next roles by their global frequency:

$$P_{\text{uni}}(j) = \frac{N(j)}{\sum_{m=1}^{K} N(m)}. \tag{3}$$

The *bigram* prior is the first context-dependent model and corresponds to a first-order Markov assumption over role transitions:

$$P_{\text{bi}}(j \mid i) = \frac{N(i \to j)}{\sum_{m=1}^{K} N(i \to m)}. \tag{4}$$

The *trigram* prior extends this to a second-order Markov model by conditioning on the previous two roles:

$$\widehat{P}_{\text{tri}}(j \mid p, i) = \frac{N(p, i \to j)}{\sum_{m=1}^{K} N(p, i \to m)}. \tag{5}$$

Higher-order n-grams suffer from sparsity when contexts are rare. We therefore use standard interpolation/backoff from language modeling [22], [23] by mixing trigram and bigram distributions:

$$P_{\text{mix}}(j \mid p, i) = (1 - \lambda)\widehat{P}_{\text{tri}}(j \mid p, i) + \lambda P_{\text{bi}}(j \mid i), \tag{6}$$

where, $\lambda \in [0, 1]$ controls the reliance on shorter contexts. We treat λ as a tunable hyperparameter selected based on performance on the validation split (Section VI).

C. Transformer Flow Model

While n-gram priors capture local dependencies, UI navigation can depend on longer histories (e.g., multi-step onboarding or configuration flows). To model longer-range context, we train a causal Transformer [24] over role sequences to estimate the flow prior $p_\theta(c_{t+1} \mid c_{1:t})$.

We formulate next-role prediction as an autoregressive sequence modeling problem over discrete role IDs. Given a role sequence $(c_1, \ldots, c_T)$, we construct training examples by prepending a beginning-of-sequence token and predicting each next role from its preceding prefix. For a prediction step at position t, the input prefix is:

$$x_{1:t} = (\texttt{[BOS]}, c_1, \ldots, c_t), \tag{7}$$

and the target is the next role c_{t+1}. For batching, prefixes are truncated to a fixed maximum context length and right-padded with a special `[PAD]` token.

Each input token is mapped to a learned token embedding and combined with a learned positional embedding. The resulting sequence is processed by a Transformer encoder with a

causal self-attention mask, so that the representation at position t depends only on the observed prefix $x_{1:t}$. Let h_t denote the contextual representation of the last non-padding token in the prefix. The model predicts the next role by applying a linear output layer followed by a softmax over the K role IDs:

$$p_\theta(c_{t+1} = j \mid c_{1:t}) = \mathrm{softmax}(Wh_t + b)_j. \tag{8}$$

Thus, the Transformer defines a conditional next-role distribution from the encoded prefix representation.

We train the model using next-token prediction over role sequences. For each prefix-target pair $(c_{1:t}, c_{t+1})$, the objective is to maximize the model's conditional probability for the observed next role in Eq. 8, or equivalently to minimize the cross-entropy loss:

$$\mathcal{L} = -\sum_{t=1}^{T-1} \log p_\theta(c_{t+1} \mid c_{1:t}). \tag{9}$$

This objective directly corresponds to learning the flow prior in Eq. 1, since the model is trained to assign high probability to the observed next role given the preceding role sequence.

To improve generalization to unseen applications, we apply regularization through dropout, weight decay, and gradient clipping. Model selection is performed using early stopping based on Recall@10 computed on a held-out validation split constructed at the application level. Model hyperparameters (e.g., context length, depth, width) and early-stopping settings are reported in Section VI.

VI. Experimental Setup

This section describes the dataset instantiation, preprocessing pipeline, multimodal representation construction, split design, model configurations, and evaluation protocol used in all experiments.

We use the RICO mobile UI interaction corpus, in which each trace is an ordered sequence of screen observations from a single application session. Each screen observation contains 1) a screenshot image and 2) a structured view hierarchy (UI tree) with widget metadata and visible text [1], [8].

A. Preprocessing and Modality Alignment

We apply the preprocessing pipeline deterministically. A screen observation is retained only if both the screenshot and its corresponding view-hierarchy JSON are present and parsable. Alignment is performed using a derived screen identifier from the file structure and metadata.

Each view hierarchy is converted into the deterministic textual representation described in Section IV-A. In preprocessing, this representation is generated by recursively traversing the hierarchy JSON in a fixed depth-first order, extracting widget identifiers and associated textual attributes, removing duplicate `(class, text)` entries within a screen, and normalizing text by trimming whitespace and removing empty strings before SBERT encoding.

For implementation robustness, the serialized hierarchy is capped at 200 entries. This cap affects only 20 of 66,261 screens (0.03%), indicating that truncation has negligible influence on the analyzed dataset.

We also compute a near-duplicate diagnostic based on embedding similarity. Using a strict threshold (top-1 cosine similarity ≥ 0.99995), 5,868 out of 66,261 screens (8.9%) are identified as near-duplicates, consistent with repeated templates and nearly identical screens within traces. This diagnostic motivates application-wise split design but is not used as an explicit filter in the main pipeline.

B. Multimodal Embedding Construction

After extracting CLIP embeddings for all 66,261 screenshots and SBERT embeddings for their corresponding serialized view hierarchies, we identified that five screenshots lacked usable hierarchy representations, resulting in an aligned multimodal intersection of 66,256 screens. We use 768-dimensional CLIP (ViT-L/14) features and 384-dimensional SBERT (MiniLM) features. After per-modality ℓ_2 normalization, the two representations are concatenated to form a 1152-dimensional fused embedding for each aligned screen.

C. Splits and Generalization Protocols

To prevent leakage and measure generalization to unseen applications, we split by application: all traces of an application belong to exactly one split. We use an 80/20 app-wise train/test split. In our snapshot, 1,503 apps are held out for testing (corresponding to the 20% test partition). For Transformer model selection, the training apps are further divided into an inner training set and an inner validation set, where the inner validation set is used for early stopping.

In addition to this app-wise train/test evaluation, we also assess out-of-domain generalization using a leave-one-category-out protocol. In this setting, all applications from one category are held out for testing, and the model is trained on applications from the remaining categories. This protocol uses 26 categories, restricted to those with at least 50 applications each. For each held-out category, we train on all other categories and evaluate on the held-out category, reporting macro-averaged performance across categories.

D. Models and Training

Table II summarizes the key model configurations and training settings used across all experiments. We evaluate unigram, bigram, and trigram flow priors on role sequences. For the trigram model, we apply trigram–bigram interpolation/backoff (Eq. 6) following established practice in n-gram modeling [22], [23]. The interpolation weight λ is treated as a hyperparameter and selected on the validation split using Recall@10 on the trigram-subset.

We also train a causal (autoregressive) Transformer for next-role prediction [24]. Each role ID is treated as a token, and we add special tokens `[BOS]` and `[PAD]`. For a role vocabulary of size K, the full token vocabulary therefore has size $K + 2$. Prefixes are clipped to at most the chosen context length `ctx` by keeping the most recent context, and shorter prefixes are right-padded for batching. The model uses learned

token embeddings, learned positional embeddings, and masked self-attention so that each position can attend only to itself and earlier tokens. The next-role distribution is predicted from the hidden state at the last non-padding position in the prefix.

Our default Transformer architecture uses embedding dimension $d = 256$, 4 attention heads, and 4 masked self-attention layers. We optimize cross-entropy loss for next-role prediction using AdamW, apply gradient clipping with maximum norm 1.0, and train with batch size 256. Unless otherwise stated, the random seed is fixed to 42 and training is allowed to continue for up to 200 epochs with early stopping. Additional implementation details are provided in Appendix B.

For model selection, we use early stopping based on validation Recall@10 on the trigram-subset, where the validation split is constructed at the application level and is disjoint from both the training and test applications. We use this ranking-based criterion because the main task is next-role retrieval rather than calibrated probability estimation, and because preliminary experiments showed that held-out Recall@10 typically peaks early while training loss can continue decreasing. We use patience $= 8$ epochs and reset patience only when validation Recall@10 improves by more than 10^{-4}.

For the controlled context-length sweep at $K = 40$, we evaluate $\texttt{ctx} \in \{4, 8, 16, 32\}$ while keeping the remaining settings fixed, and we report the best checkpoint under the above validation criterion. For the reported multi-K Transformer runs, unless otherwise specified, we use $\texttt{ctx}= 16$, learning rate 3×10^{-5}, dropout 0.3, weight decay 0.1, batch size 256, and an application-level inner validation ratio of 0.2.

E. Evaluation Protocol

Given a test trace converted into a role sequence, we create prediction instances at each position by using the observed prefix and predicting the next role. We report Recall@k for $k \in \{1, 3, 5, 10, 20\}$.

We report two evaluation modes: 1) *all edges*, which includes all transitions with at least one preceding role, and 2) the *trigram-subset*, which restricts evaluation to transitions with at least two preceding roles for fair comparison with trigram-based methods. Unless otherwise specified, all models are evaluated on the same set of trigram-subset transition edges for fair comparison.

We evaluate atypical transition detection by converting next-step prediction into a sampled-candidate ranking problem with negative alternatives. For each true transition, we form one positive candidate (the observed next role) and $N_{\text{neg}} = 10$ negative candidates by replacing the next role with roles sampled uniformly at random from the role vocabulary $\{1, \ldots, K\} \setminus \{c_{t+1}\}$. Special tokens such as `[PAD]` and `[BOS]` are excluded from this sampling. We use a fixed random seed for reproducibility. This protocol keeps the candidate set size fixed across examples and models, while providing a lightweight approximation of anomaly discrimination without requiring exhaustive scoring over all possible next roles. Because the negatives are sampled uniformly at random, the task measures discrimination against random incorrect alternatives rather than against hardest or semantically nearest negatives. All candidates are scored using the model's assigned probability for the next role under the same observed context. We report AUROC and Precision–Recall metrics (including Average Precision) to account for the resulting class imbalance.

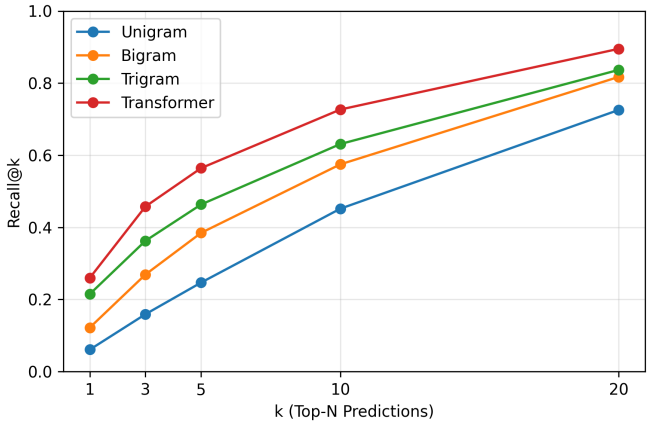

Fig. 3. Recall@k curves for next-step prediction on applications in the test split ($K = 40$), comparing unigram, bigram, trigram, and Transformer priors.

Accordingly, these metrics should be interpreted as measuring relative ranking performance under sampled alternatives, not as a direct estimate of real-world anomaly-detection accuracy under unconstrained deployment conditions.

VII. Results

We report results for the two tasks defined in Section III: 1) next-step suggestion via Recall@k and 2) atypical transition detection via ranking/classification metrics under negative sampling. Unless otherwise stated, results use the primary cross-app protocol (Section VI-C) and matched-edge evaluation for fair comparison.

A. Task 1: Next-Step Prediction Performance

Table III compares unigram, bigram, trigram, and the Transformer flow prior on the same matched set of 6,058 transition edges. The Transformer achieves the strongest Recall@k at every reported cutoff. Relative to the trigram baseline, the absolute improvements are +0.0449 at R@1, +0.0952 at R@3, +0.1009 at R@5, +0.0957 at R@10, and +0.0585 at R@20. Thus, the improvement is not simply monotonic in k; rather, the largest absolute gains occur in the shortlist range from R@3 to R@10, which is particularly relevant for shortlist-based design-assistance settings.

Fig. 3 visualizes the Recall@k curves and makes the differences in top-k ranking behavior easier to compare than the tabulated values alone.

Additional Transformer ablations over context length and regularization are reported in Appendix A.

To assess robustness to neural training randomness, we repeated the main $K = 40$ Transformer experiment with three random seeds while keeping the data split and model recipe fixed. As shown in Table IV, performance remains stable across seeds, with low standard deviation at all reported

TABLE II. KEY EXPERIMENTAL SETTINGS USED ACROSS MODELS

Component	Setting
Role vocabulary size	$K \in \{20, 40, 80, 120, 200\}$ (reported per experiment)
Duplicate collapsing	collapse consecutive identical roles; drop traces < 2
n-gram backoff	trigram–bigram interpolation (Eq. 6)
n-gram λ	tuned on validation split (maximizing Recall@10 on matched subset)
Transformer architecture	causal/autoregressive Transformer with masked self-attention
Transformer width/depth	$d = 256$, 4 heads, 4 layers
Transformer objective	cross-entropy next-role prediction
Transformer optimizer	AdamW
Transformer context length	ctx $\in \{4, 8, 16, 32\}$ (controlled sweep at $K = 40$)
Model selection	early stop on validation Recall@10 (trigram-subset), best checkpoint chosen
Training epochs	up to 200 with early stopping
Gradient clipping	max norm $= 1.0$
Transformer reported multi-K recipe	ctx $= 16$, lr $= 3 \times 10^{-5}$, dropout $= 0.3$, weight decay $= 0.1$, batch size $= 256$
Random seed	42 for the primary reported run; additional multi-seed robustness results are reported in Appendix E
Anomaly protocol	primary setting uses $N_{\text{neg}} = 10$; see App. F and App. G

TABLE III. RECALL@K FOR NEXT-STEP PREDICTION

Model	R@1	R@3	R@5	R@10	R@20
Unigram	0.0609	0.1591	0.2469	0.4518	0.7257
Bigram	0.1220	0.2692	0.3848	0.5748	0.8173
Trigram	0.2149	0.3627	0.4635	0.6311	0.8367
Transformer	**0.2598**	**0.4579**	**0.5644**	**0.7268**	**0.8952**

Recall@k for next-step prediction on unseen test applications with $K = 40$. All models are evaluated on the same 6,058 matched transition edges for fair comparison.

cutoffs: 0.2671 ± 0.0066 at R@1, 0.4691 ± 0.0097 at R@3, 0.5817 ± 0.0152 at R@5, 0.7436 ± 0.0146 at R@10, and 0.9032 ± 0.0071 at R@20. This indicates that the Transformer advantage is not tied to a favorable initialization, but persists across independent runs. Seed-specific results are provided in Appendix E.

1) Modality ablation at $K = 40$: To assess the contribution of each screen-representation modality, we repeated the $K = 40$ pipeline using three alternatives for screen-role induction: CLIP-only visual embeddings, SBERT-only hierarchy-text embeddings, and the fused CLIP+SBERT representation used in the main experiments. For each representation, we re-ran clustering, trace-to-role conversion, and downstream next-step modeling under the same application-level split and evaluation protocol. Table V shows that no single modality dominates uniformly across all models and recall cutoffs. For the Transformer, the fused representation yields the strongest top-ranked performance at $R@5$ and is effectively tied with SBERT-only at $R@1$, while CLIP-only slightly outperforms fused at $R@10$. This pattern suggests that fusion is best interpreted as a balanced representation choice rather than as a universally superior modality under all retrieval cutoffs.

We use $K = 40$ as the primary operating point in the remainder of this section. This choice is not based on raw next-step recall alone, since smaller values of K make the prediction problem easier by construction. Instead, $K = 40$ was selected in Section IV-C using clustering stability and fragmentation as the primary criteria, with downstream prediction used as a secondary check that the resulting role vocabulary remains predictive on held-out applications. Additional low-granularity diagnostics at $K = 10$ and $K = 15$ show that, although smaller values yield higher raw Recall@k, they also produce substantially coarser role vocabularies with much larger cluster sizes. We therefore use $K = 20$ as the coarse-grained lower bound in the downstream sweep and $K = 40$ as the representative operating point because it provides finer role granularity while still maintaining strong downstream performance.

Table VI reports next-step prediction performance across cluster granularities $K \in \{20, 40, 80, 120, 200\}$ under the same unseen test-application evaluation protocol; additional low-granularity results for $K = 10$ and $K = 15$ are reported in the Appendix. At $K = 20$, the $R@20$ column is saturated (1.0000 for all models) because the evaluation cutoff equals the role vocabulary size. This is a limitation of the current evaluation protocol at this granularity: with a coarse role space, $R@20$ no longer provides discriminative evidence about model quality and instead becomes a trivial upper bound. In other words, the task at this granularity is too coarse for $R@20$ to be informative under the current setup. Accordingly, the meaningful comparisons at $K = 20$ are $R@1$, $R@3$, $R@5$, and $R@10$.

As expected, raw Recall@k decreases as K increases, because the prediction task becomes harder when the role vocabulary is larger and the transition distribution becomes sparser. Across all K, the Transformer consistently outperforms the n-gram baselines.

To test out-of-domain generalization beyond application identity, we additionally evaluate leave-one-category-out with $K = 40$. Macro-averaged results over 26 held-out categories are shown in Table VII. The Transformer achieves the best macro-averaged Recall@k across all reported cutoff values, indicating stronger average category-level generalization than the n-gram baselines.

TABLE IV. MULTI-SEED TRANSFORMER NEXT-STEP PREDICTION RESULTS ($K = 40$)

Model	R@1	R@3	R@5	R@10	R@20
Transformer (3 seeds)	0.2671 ± 0.0066	0.4691 ± 0.0097	0.5817 ± 0.0152	0.7436 ± 0.0146	0.9032 ± 0.0071

Results are reported as mean $\pm$ standard deviation over three random seeds on unseen test applications.

TABLE V. MODALITY ABLATION ACROSS NEXT-STEP MODELS AT $K = 40$

Representation	Model	$R@1$	$R@5$	$R@10$
CLIP-only	Unigram	0.0668	0.2974	0.4797
Fused	Unigram	0.0950	0.3158	0.4958
SBERT-only	Unigram	0.0824	0.3042	0.4586
CLIP-only	Bigram	0.1198	0.4031	0.6127
Fused	Bigram	0.1455	0.4227	0.6104
SBERT-only	Bigram	0.1403	0.3946	0.5704
CLIP-only	Trigram	0.1967	0.4659	0.6448
Fused	Trigram	0.2199	0.4714	0.6367
SBERT-only	Trigram	0.2245	0.4750	0.6267
CLIP-only	Transformer	0.2224	0.5303	**0.7300**
Fused	Transformer	**0.2465**	**0.5629**	0.7216
SBERT-only	Transformer	0.2464	0.5525	0.6908

Each representation is used to induce screen roles, after which the same downstream application-level split and evaluation protocol are applied.

B. Task 2: Atypical Transition Detection

We evaluate atypical transition detection by ranking the true next role against $N_{\text{neg}} = 10$ randomly sampled incorrect roles per transition (Section VI-E). On the unseen test set, the Transformer achieves strong ranking performance, with AUROC = 0.83 and Average Precision (AP) = 0.42 under this protocol. These values indicate how well the model ranks observed transitions against sampled random alternatives under a fixed candidate set, and should not be interpreted as direct evidence of real-world anomaly-detection performance.

We compare n-gram baselines and the Transformer under an identical evaluation protocol: The same 6,058 test transitions, fixed negative sampling with $N_{\text{neg}} = 10$ (positive rate ≈ 0.0909), and the same sampled negatives reused across models. Table VIII reports AUROC, Average Precision (AP), and recall at a precision-constrained operating point (precision ≥ 0.8).

We also examined sensitivity to the number of sampled negatives in the anomaly protocol. Table IX shows that AUROC remains essentially unchanged across $N_{\text{neg}} \in \{5, 10, 20, 50\}$, varying only from 0.8317 to 0.8328. This suggests that the Transformer's relative ranking quality is robust to the size of the sampled candidate set. In contrast, Average Precision and recall at precision ≥ 0.8 decrease substantially as N_{neg} increases. This is expected, since larger N_{neg} makes the sampled task more imbalanced and pushes the high-precision operating point toward increasingly conservative thresholds. The result therefore strengthens the interpretation that the Transformer's anomaly advantage is robust as a ranking result, while threshold-dependent operating points remain sensitive to the difficulty of the sampled protocol. Additional results are reported in Appendix F.

To test whether the sampled-candidate protocol materially overstates anomaly-ranking performance, we additionally re-evaluated the main Transformer checkpoint against the full role vocabulary at $K = 40$ on the same 6,058 matched test transitions. Under this stricter setting, AUROC remains essentially unchanged (0.8320, versus 0.8323 under sampled negatives with $N_{\text{neg}} = 10$), while Average Precision decreases from 0.4231 to 0.1943 and recall at precision ≥ 0.8 decreases from 0.0568 to 0.0028. This indicates that the Transformer's relative ranking quality is robust beyond the sampled-negative setting, but that threshold-dependent anomaly screening becomes substantially more conservative when each context is evaluated against the full vocabulary.

To examine threshold behavior under different precision constraints, Table X reports thresholds selected to maximize recall while maintaining a target precision level. The strictest settings (e.g., target precision 0.95 or 0.90) achieve extremely low recall, indicating that these thresholds should be interpreted as highly conservative operating points rather than broad anomaly-screening regimes.

These results show that, although the Transformer provides the strongest overall ranking performance under the sampled evaluation protocol, very high precision comes at a substantial cost in recall. At target precision 0.95 and 0.90, only a very small fraction of atypical transitions is recovered. Lower precision targets such as 0.70, 0.60, or 0.50 provide substantially higher recall, but this comes at the expected cost of admitting more false positives.

Accordingly, Table X should be interpreted as illustrating the precision–recall trade-off under the fixed negative-sampling protocol rather than as establishing a deployment threshold for real-world anomaly detection.

Fig. 4 visualizes the full precision–recall curve and highlights the operating points in Table X.

C. Qualitative Analysis of Detected Redirect Failures

To complement quantitative ranking metrics, we report qualitative examples of high-confidence anomalies detected by the Transformer. Examples are selected from cases in which the observed transition is assigned low probability and there is a large margin between its probability and that of the top predicted continuation.

Fig. 5 shows an example from the KAYAK application. After a system permission request, the observed transition redirects the user to a hotel sharing interface that is unrelated to the permission context. The model assigns low probability to the observed continuation ($p_{\text{obs}} \approx 0.0035$) while ranking an account login/linking screen as the most likely continuation ($p_{\text{exp}} \approx 0.34$).

Fig. 6 shows an example from a transportation-related application. From a tariff configuration screen, the observed transition navigates to a driver configuration page without an apparent authentication step. The model instead predicts a login screen as the most likely continuation, assigning substantially higher probability than the observed transition.

TABLE VI. NEXT-STEP PREDICTION PERFORMANCE ACROSS CLUSTER GRANULARITIES

K	Model	R@1	R@3	R@5	R@10	R@20
20	Unigram	0.1176	0.2837	0.4146	0.6978	**1.0000**
20	Bigram	0.1588	0.3606	0.5083	0.7589	**1.0000**
20	Trigram	0.2851	0.4570	0.5830	0.7915	**1.0000**
20	Transformer	**0.3237**	**0.5442**	**0.6528**	**0.8365**	**1.0000**
40	Unigram	0.0609	0.1591	0.2469	0.4518	0.7257
40	Bigram	0.1220	0.2692	0.3848	0.5748	0.8173
40	Trigram	0.2149	0.3627	0.4635	0.6311	0.8367
40	Transformer	**0.2598**	**0.4579**	**0.5644**	**0.7268**	**0.8952**
80	Unigram	0.0440	0.1187	0.1690	0.2897	0.4743
80	Bigram	0.0993	0.2175	0.3015	0.4501	0.6365
80	Trigram	0.1567	0.2863	0.3638	0.4933	0.6655
80	Transformer	**0.2433**	**0.4023**	**0.4928**	**0.6284**	**0.7757**
120	Unigram	0.0337	0.0860	0.1273	0.2144	0.3632
120	Bigram	0.0876	0.1933	0.2648	0.3892	0.5416
120	Trigram	0.1217	0.2359	0.3094	0.4193	0.5685
120	Transformer	**0.2030**	**0.3528**	**0.4360**	**0.5588**	**0.6953**
200	Unigram	0.0171	0.0493	0.0770	0.1384	0.2493
200	Bigram	0.0734	0.1580	0.2203	0.3156	0.4436
200	Trigram	0.0959	0.1882	0.2497	0.3416	0.4621
200	Transformer	**0.1783**	**0.3095**	**0.3811**	**0.4956**	**0.6107**

Next-step prediction performance across cluster granularities. All results are evaluated on unseen test applications using cumulative Recall@k.

TABLE VII. LEAVE-ONE-CATEGORY-OUT RESULTS

Model	R@1	R@3	R@5	R@10
Unigram	0.0640	0.1604	0.2637	0.4631
Bigram	0.1217	0.2770	0.3891	0.5693
Trigram	0.1963	0.3590	0.4596	0.6283
Transformer	**0.2105**	**0.4115**	**0.5257**	**0.6985**

Leave-one-category-out

macro-average results over 26 held-out categories ($K = 40$), evaluated on the trigram-subset.

TABLE VIII. ATYPICAL TRANSITION DETECTION

Model	AUROC ↑	AP ↑	Recall @ Precision ≥ 0.8 ↑
Unigram	0.665	0.149	0.0000
Bigram	0.746	0.242	0.0000
Trigram	0.771	0.336	0.0154
Transformer	0.832	0.423	0.0568

Atypical

transition detection under a fixed protocol (same 6,058 transitions; fixed negative sampling with $N_{\text{neg}} = 10$; positive rate ≈ 0.0909). We report AUROC, Average Precision (AP), and recall at precision ≥ 0.8.

TABLE IX. SENSITIVITY OF TRANSFORMER ATYPICAL-TRANSITION METRICS TO THE NUMBER OF SAMPLED NEGATIVES N_{neg} ($K = 40$)

N_{neg}	AUROC ↑	AP ↑	Recall@Precision ≥ 0.8 ↑
5	0.8317	0.5626	0.1720
10	0.8323	0.4231	0.0568
20	0.8319	0.2957	0.0102
50	0.8328	0.1654	0.0012

TABLE X. PRECISION-RECALL TRADE-OFFS FOR TRANSFORMER-BASED ATYPICAL TRANSITION DETECTION

Target Precision	Threshold	Achieved Precision	Recall
0.95	0.9114	1.0000	0.00033
0.90	0.8618	0.9231	0.00396
0.70	0.3774	0.7003	0.1354
0.60	0.1904	0.6000	0.2565
0.50	0.1030	0.5000	0.3760

Thresholds are chosen to maximize recall while satisfying a target precision constraint. Very high target precision yields highly conservative operating points with limited recall.

VIII. DISCUSSION

Our results support the central hypothesis of this work: after abstracting screens into reusable role IDs, mobile UI navigation exhibits domain-independent regularities that can be learned from large-scale traces and transferred to unseen applications and categories. We discuss implications for next-step suggestion and atypical transition detection, analyze the effect of role granularity, and outline limitations and future directions.

A. Next-Step Prediction on Unseen Applications

Across unseen test applications, the Transformer flow prior consistently outperforms count-based n-gram baselines, with the largest absolute gains occurring in the shortlist range from R@3 to R@10 (Table III, Fig. 3). For a fixed shortlist size, this means that the Transformer more often places the true next role among the top-ranked candidates, which is compatible with shortlist-based design-assistance settings in which a small set of plausible continuations is presented to the user.

A plausible interpretation is that role sequences contain recurring multi-step patterns (e.g., transitions from entry to browsing to detail to confirmation states) that benefit from sequence models able to condition on longer histories, whereas n-gram baselines are limited to fixed-order local context. However, the present experiments establish the empirical advantage of the Transformer rather than the exact mechanism responsible for it.

The sensitivity study across K (Table VI) further clarifies the role of abstraction. As K increases, the next-role classification problem becomes harder because the role vocabulary is larger and the transition distribution becomes sparser. This trend affects all models, but the degradation is steeper for frequency-based baselines that rely directly on repeated transition counts. In contrast, the Transformer maintains stronger performance across the full range of tested granularities, indicating that it is better able to preserve useful ranking quality as the role abstraction becomes finer.

Throughout the main results we report $K = 40$ as the primary operating point. This choice is not based on raw next-step recall alone, since smaller values of K make the prediction task easier by construction. Instead, $K = 40$ was selected as a practical trade-off between representational granularity,

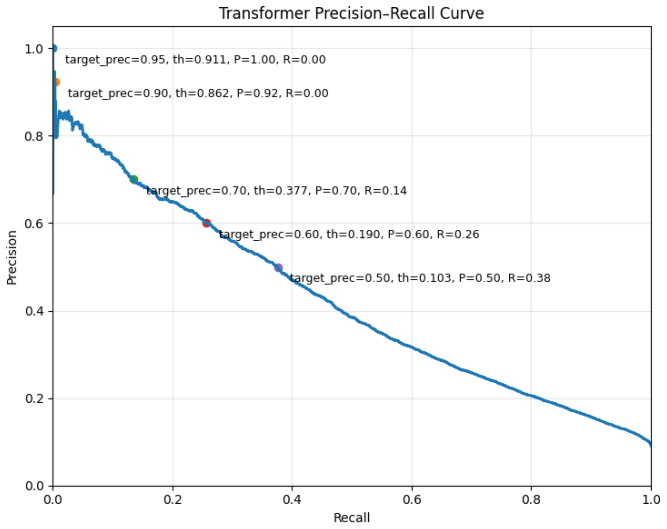

Fig. 4. Precision-recall curve for Transformer-based atypical transition detection under negative sampling. Markers correspond to selected precision-constrained operating points.

clustering stability, and downstream predictive performance, supported by the diagnostics in Section IV-C together with the sensitivity analysis in Table VI. We nevertheless report results across multiple values of K and treat role granularity as a tunable design parameter rather than a fixed constant.

B. Generalization Beyond Applications: Cross-Category Behavior

The leave-one-category-out protocol tests a more challenging distribution shift than cross-application splits, since the held-out category may involve different UI conventions, vocabulary, and navigational patterns. The n-gram results (Table VII) show that the role abstraction retains predictive value even under this stronger shift. The Transformer further improves macro-averaged Recall@k under the same protocol, providing evidence of stronger average generalization across held-out categories than the count-based baselines.

At the same time, these results should be interpreted conservatively. Categories in RICO are not perfectly disjoint in UI structure, and some categories may still share common design conventions such as list-detail patterns or authentication flows. The full per-category breakdown confirms that the macro-average masks meaningful heterogeneity rather than uniform behavior across held-out categories. In the present experiments, Transformer R@10 ranges from 0.5446 on *Sports* and 0.5843 on *Food & Drink* to 0.7666 on *Education*, 0.7455 on *Weather*, and 0.7452 on *Beauty*. Thus, the leave-one-category-out result should be read as evidence of strong average cross-category generalization together with substantial variation in category difficulty. Full per-category results are reported in Appendix H.

A plausible explanation for the weaker held-out results on *Sports* and *Food & Drink* is that these categories often involve more heterogeneous and content-driven flows than the strongest held-out categories. Sports applications may combine news, live scores, video, team pages, betting- or fantasy-like features, and account flows within the same app family, while Food & Drink applications may mix restaurant discovery, menus, ordering, reservation, delivery, and loyalty flows. Such mixtures can produce more branching continuations from visually similar intermediate states and therefore reduce next-role predictability under a shared role vocabulary. By contrast, stronger categories such as Education, Weather, and Beauty may rely more heavily on recurring template-like flows such as onboarding, settings, browsing, and detail views, which transfer more readily across applications. These explanations are interpretive rather than directly annotated in the present dataset, but they are consistent with the heterogeneity visible in the full leave-one-category-out breakdown.

C. Interpreting Atypical Transition Detection Metrics

The atypical-transition results in this study should be interpreted within the scope of the evaluation protocol. Specifically, the task is defined through fixed negative sampling, so the reported metrics measure how well each model ranks observed transitions against sampled alternatives under this controlled setting. They therefore support comparison of relative ranking performance across models, but do not by themselves establish real-world anomaly-detection effectiveness under unconstrained deployment conditions.

In particular, the protocol does not test whether low-probability transitions correspond to naturally occurring defects in deployment, since sampled negatives may differ substantially from realistic alternative continuations.

At the same time, an additional full-vocabulary re-evaluation at $K = 40$ shows that AUROC remains nearly unchanged relative to the sampled protocol, whereas Average Precision and precision-constrained recall drop substantially, indicating that sampled negatives preserve relative ranking comparisons reasonably well but understate the difficulty of strict high-precision anomaly screening. Even with this stricter full-vocabulary comparison, the evaluation remains an offline atypical-transition ranking task rather than a direct benchmark of naturally occurring deployment anomalies.

Within this fixed protocol, the Transformer achieves the strongest ranking performance among the compared models (Table VIII), indicating better discrimination between observed transitions and sampled negative alternatives than the n-gram baselines. Relative to the trigram baseline, this result suggests that atypical-transition scoring may benefit from information beyond short local context, although the present experiments do not directly establish the underlying mechanism.

At the same time, the precision-constrained analysis (Table X, Fig. 4) highlights an important limitation. At very high precision targets such as 0.95 or 0.90, recall becomes extremely low. These operating points should therefore not be interpreted as evidence of broad practical usefulness; rather, they show that only a very small fraction of positives can be recovered when false alarms are tightly constrained. Lower precision targets provide substantially higher recall, but this comes at the expected cost of admitting more false positives.

Overall, the anomaly results are best understood as evidence that the Transformer provides the strongest relative ranking performance under the sampled evaluation protocol, while also revealing clear practical limitations in strict high-precision regimes. A more realistic assessment of deployment

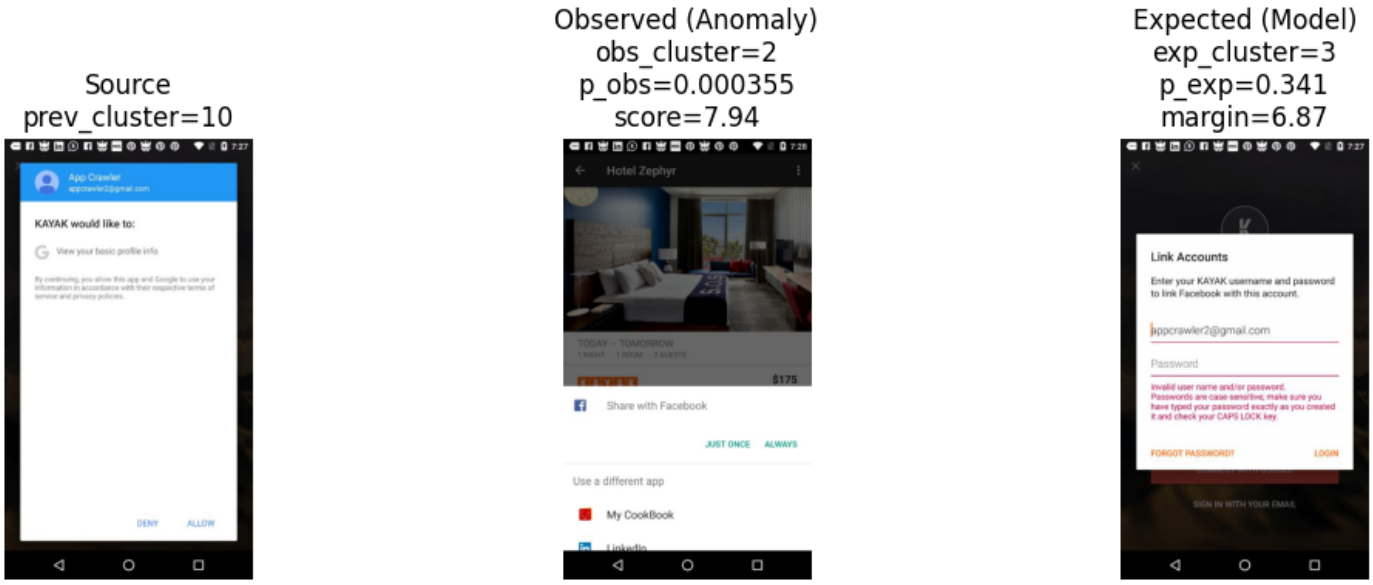

Fig. 5. Detected redirect failure in the KAYAK application. After a permission request (left), the observed transition leads to an unrelated sharing interface (middle), while the model predicts an account login screen as the expected continuation (right).

Fig. 6. Detected configuration flow anomaly. From a tariff configuration screen (left), the application navigates to a driver settings page without authentication (middle). The model instead predicts a login screen as the expected continuation (right).

utility would require evaluation settings that go beyond sampled negatives and better reflect naturally occurring atypical transitions.

D. Qualitative Inspection of High-Scoring Transitions

The qualitative case studies (Fig. 5 and 6) are interpreted as illustrative examples rather than as a general validation of the anomaly score. In the presented cases, high anomaly scores coincide with transitions that appear questionable or unusual from a design perspective, while the model assigns substantially higher probability to an alternative continuation that is more consistent with the preceding flow context (e.g., an authentication or account-linking step). These examples therefore suggest that the score can highlight potentially design-relevant irregularities in at least some cases, but they do not establish that this holds broadly beyond the illustrated examples.

The contrast between the observed continuation and the model-preferred alternative can nevertheless provide a qualitative basis for inspection. A designer can examine both what actually occurred and what the model ranked as more probable under the learned flow prior. This comparison may

help interpret why a transition was flagged, although the present qualitative evidence is not sufficient to claim general diagnostic usefulness.

A limitation is that qualitative selection can be biased toward visually obvious failures. To mitigate this, future qualitative evaluation should include a structured sampling strategy (e.g., top anomalies stratified by application/category, plus a random sample of mid-score transitions) together with human annotation to estimate how often low-probability transitions correspond to true flow defects versus legitimate but rare paths.

E. Role Granularity and the Choice of K

The experiments do not support selecting K from next-step prediction accuracy alone. Smaller values of K produce higher Recall@k, but also correspond to coarser role vocabularies in which more heterogeneous screens are merged together. Conversely, larger values of K preserve finer distinctions but lead to increased sparsity and weaker transition modeling performance.

On this basis, we use $K = 40$ as the main operating point in the study. This choice is supported by two considerations grounded in the reported experiments: 1) downstream predictive behavior on held-out applications, where $K = 40$ retains strong next-step prediction performance without relying on the overly coarse abstraction induced by smaller K values; and 2) clustering diagnostics in Section IV-C, where $K = 40$ shows favorable stability while avoiding the stronger fragmentation observed at larger values of K. We therefore treat $K = 40$ not as a universally "natural" constant, but as a practically selected operating point supported by the present evidence.

More generally, the experiments suggest that role granularity should be treated as a tunable design parameter whose preferred value depends on the balance one wants between semantic specificity and statistical reliability.

F. Limitations and Future Work

This study has several limitations. First, the primary atypical-transition evaluation uses negative sampling, which provides a controlled quantitative protocol but does not fully capture real-world anomalies. We complement this with a full-vocabulary re-evaluation at $K = 40$, but both settings remain offline ranking evaluations rather than direct measurements of naturally occurring deployment failures. In practice, unusual transitions may depend on latent application or user state, such as authentication status, granted permissions, network conditions, or prior interaction history, none of which is explicitly modeled in the current setup. As a result, the anomaly results should be interpreted as evidence of ranking quality under controlled candidate-comparison settings rather than as a complete evaluation of real deployment conditions.

Second, we added four robustness analyses to reduce dependence on a single reported configuration: multi-seed evaluation for the Transformer, sensitivity analysis with respect to the number of sampled negatives N_{neg}, an exhaustive full-vocabulary evaluation at $K = 40$, and a full per-category breakdown under leave-one-category-out. Together, these analyses show that the main next-step gains are stable across random seeds, that anomaly-ranking performance remains stable in AUROC across different sampled-negative settings, that exhaustive comparison is substantially harsher than sampled evaluation while still preserving useful ranking quality, and that cross-category generalization remains strong on average despite substantial heterogeneity across held-out categories. The remaining limitation is therefore not the absence of these checks, but the fact that anomaly evaluation is still defined primarily through sampled negatives rather than naturally occurring deployment anomalies. An additional full-vocabulary re-evaluation at $K = 40$ shows that AUROC is largely unchanged relative to the sampled protocol, but that Average Precision and precision-constrained recall decrease substantially, so the sampled setting appears to preserve relative ranking comparisons better than it preserves realistic high-precision screening difficulty.

Beyond these limitations, several research extensions are promising. One direction is to design more realistic anomaly benchmarks that better account for latent state, for example by constructing negatives within matched application/session contexts or by incorporating human-labeled subsets of truly atypical transitions. Another is to expand the comparison set beyond n-gram baselines and a causal Transformer, for example by evaluating recurrent models, structured probabilistic sequence models, or retrieval-augmented priors, in order to better understand which sequence-modeling assumptions are most suitable for UI flow data. Finally, because the current role abstraction is unsupervised and intentionally label-free, an important direction is to generate lightweight role descriptors (e.g., by summarizing common hierarchy tokens or nearest-neighbor exemplars) to improve interpretability and debugging without requiring manual role annotation. While this qualitative audit improves interpretability, it does not replace a formal human evaluation of cluster semantics. A larger-scale annotated study remains an important direction for future work.

IX. Conclusion

We presented a role-based formulation of mobile UI navigation in which screens are mapped to discrete, unsupervised *screen roles* using multimodal representations and clustering, allowing raw interaction traces to be modeled as role sequences across applications. This abstraction supports cross-application sequence modeling while preserving enough structure to study downstream navigation behavior.

Across unseen test applications, both n-gram baselines and a causal Transformer capture nontrivial regularities in role transitions, with the Transformer consistently achieving the strongest next-step prediction performance. Under the fixed negative-sampling protocol used for atypical-transition evaluation, the Transformer also achieves the strongest ranking performance among the compared models. Sensitivity analysis across role granularities further shows that the choice of K should be treated as a practical design parameter rather than a fixed constant; in this study, $K = 40$ serves as a representative operating point supported by clustering diagnostics and downstream predictive behavior.

The qualitative examples illustrate that some high-scoring atypical transitions can correspond to potentially design-relevant irregularities, although broader validation would require more realistic anomaly benchmarks and structured hu-

man evaluation. More generally, the results indicate that role-based abstractions provide a promising basis for studying UI flow regularities and for ranking-oriented tasks such as next-step suggestion under cross-application generalization.

The robustness analyses further indicate that these conclusions are stable across random seeds, across multiple negative-sampling settings, under a stricter full-vocabulary comparison, and across detailed held-out-category breakdowns. Future work should therefore focus less on basic stability verification and more on richer latent-state information, improved interpretability through lightweight role descriptors, and more realistic anomaly-evaluation protocols.

ACKNOWLEDGMENT

The authors would like to thank the creators of the *Rico* dataset for making it publicly available, which made this research possible.

CODE AVAILABILITY

The source code implementing the models and training procedures used in this study is available at: https://github.com/salsik/trico. The repository also includes the exact application-level split manifests used in the main experiments, including the train/validation/test app partitions for the cross-application protocol and the held-out category lists for the leave-one-category-out evaluation.

REFERENCES

[1] B. Deka, Z. Huang, C. Franzen, J. Hibschman, D. Afergan, Y. Li, J. Nichols, and R. Kumar, "Rico: A mobile app dataset for building data-driven design applications," in *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology (UIST '17)*, 2017, pp. 845–854.

[2] T. J.-J. Li, L. Popowski, T. M. Mitchell, and B. A. Myers, "Screen2vec: Semantic embedding of gui screens and gui components," *arXiv preprint arXiv:2101.11103*, 2021. [Online]. Available: https://arxiv.org/abs/2101.11103

[3] B. Wang, G. Li, X. Zhou, Z. Chen, T. Grossman, and Y. Li, "Screen2words: Automatic mobile ui summarization with multimodal learning," in *Proceedings of the 34th Annual ACM Symposium on User Interface Software and Technology (UIST '21)*, 2021.

[4] L. A. Leiva, A. Hota, and A. Oulasvirta, "Enrico: A dataset for topic modeling of mobile ui designs," in *22nd International Conference on Human-Computer Interaction with Mobile Devices and Services (MobileHCI '20 Extended Abstracts)*, 2020.

[5] L. Fan, T. Su, S. Chen, G. Meng, Y. Liu, and L. Xu, "Storydroid: Automated generation of storyboard for android apps," in *Proceedings of the 41st International Conference on Software Engineering (ICSE '19)*, 2019.

[6] X. Zhou and Y. Li, "Large-scale modeling of mobile user click behaviors using deep learning," in *Proceedings of the 15th ACM Conference on Recommender Systems (RecSys '21)*, 2021.

[7] S. Feiz, J. Wu, X. Zhang, A. Swearngin, T. Barik, and J. Nichols, "Understanding screen relationships from screenshots of smartphone applications," in *Proceedings of the 27th International Conference on Intelligent User Interfaces (IUI '22)*, 2022.

[8] B. Deka, Z. Huang, and R. Kumar, "Erica: Interaction mining mobile apps," in *Proceedings of the 29th Annual Symposium on User Interface Software and Technology (UIST '16)*, 2016, pp. 767–776.

[9] G. Hu, L. Zhu, and J. Yang, "Appflow: Using machine learning to synthesize robust, reusable ui tests," in *Proceedings of the 2018 26th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering (ESEC/FSE '18)*, 2018.

[10] J. Wu, Y.-H. Peng, X. Y. A. Li, A. Swearngin, J. P. Bigham, and J. Nichols, "Uiclip: A data-driven model for assessing user interface design," in *Proceedings of the 37th Annual ACM Symposium on User Interface Software and Technology (UIST '24)*, 2024, pp. 45:1–45:16.

[11] I. Salama, L. H. Mormille, and M. Atsumi, "Laycoder: UI layout completion with an encoder-only transformer and layout tokenizer," *International Journal of Advanced Computer Science and Applications*, vol. 17, no. 2, 2026.

[12] T. J.-J. Li, A. Azaria, B. A. Myers, and J. Nichols, "Kite: Building conversational bots from mobile apps," in *Proceedings of the 16th ACM International Conference on Mobile Systems, Applications, and Services (MobiSys '18)*, 2018.

[13] G. Li, X. Zhang, A. Swearngin, and J. Nichols, "Sift: Grounding multimodal language models for interactive ui tasks," in *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI)*, 2023.

[14] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever, "Learning transferable visual models from natural language supervision," in *Proceedings of the 38th International Conference on Machine Learning (ICML)*, 2021.

[15] N. Reimers and I. Gurevych, "Sentence-bert: Sentence embeddings using siamese bert-networks," in *Proceedings of EMNLP-IJCNLP*, 2019.

[16] D. Sculley, "Web-scale k-means clustering," in *Proceedings of the 19th international conference on World Wide Web*, 2010, pp. 1177–1178.

[17] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. VanderPlas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and É. Duchesnay, "Scikit-learn: Machine learning in python," *Journal of Machine Learning Research*, vol. 12, no. 85, pp. 2825–2830, 2011. [Online]. Available: https://www.jmlr.org/papers/v12/pedregosa11a.html

[18] P. J. Rousseeuw, "Silhouettes: A graphical aid to the interpretation and validation of cluster analysis," *Computational and Applied Mathematics*, vol. 20, pp. 53–65, 1987.

[19] D. L. Davies and D. W. Bouldin, "A cluster separation measure," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 1, no. 2, pp. 224–227, 1979.

[20] T. Calinski and J. Harabasz, "A dendrite method for cluster analysis," *Communications in Statistics*, vol. 3, no. 1, pp. 1–27, 1974.

[21] A. Strehl and J. Ghosh, "Cluster ensembles—a knowledge reuse framework for combining multiple partitions," *Journal of Machine Learning Research*, vol. 3, pp. 583–617, Dec. 2002. [Online]. Available: https://www.jmlr.org/papers/v3/strehl02a.html

[22] S. F. Chen and J. Goodman, "An empirical study of smoothing techniques for language modeling," *Computer Speech & Language*, vol. 13, no. 4, pp. 359–394, 1999.

[23] D. Jurafsky and J. H. Martin, "Speech and language processing: An introduction to natural language processing, computational linguistics, and speech recognition with language models," 2026, online manuscript released January 6, 2026. [Online]. Available: https://web.stanford.edu/~jurafsky/slp3/

[24] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

APPENDIX

This appendix provides additional experimental details and supplementary results that support the main work.

A. Transformer Ablations for Next-Step Prediction

We conducted a broad hyperparameter sweep for Transformer-based next-step prediction under the cross-application protocol, using $K = 40$ as the main operating point. In total, the sweep comprised 120 training runs. The goal of this search was to verify that the reported Transformer

results are stable under reasonable variations in sequence context and training hyperparameters, and to select the checkpoint used in the main experiments.

Across the sweep, we varied maximum context length, learning rate, dropout, and the application-level inner validation ratio. Specifically, the explored values were $\texttt{ctx} \in \{4, 8, 16, 32\}$, learning rate $\in \{3 \times 10^{-5}, 10^{-4}, 3 \times 10^{-4}\}$, dropout $\in \{0.1, 0.2, 0.3\}$, and validation ratio $\in \{0.05, 0.1, 0.2\}$. All runs used label smoothing 0.0, a maximum training budget of 200 epochs, and patience $= 8$. Table XI reports the strongest configurations from this 120-run sweep rather than listing all runs individually.

The best validation checkpoint used throughout the main experiments is obtained with context length 16, learning rate 3×10^{-5}, dropout 0.3, weight decay 0.1, batch size 256, and validation ratio 0.2, achieving a best validation score of 0.7187. Two general patterns are visible in the sweep. First, context length 16 consistently outperforms shorter context (8) and remains preferable to the longer setting (32), suggesting that moderate-length history is sufficient for most predictive gains in the role-sequence setting. Second, mild regularization improves selection performance, with dropout 0.3 and weight decay 0.1 appearing more reliably among the strongest runs than weaker regularization settings.

Because this selected checkpoint is the one used in all downstream experiments, the sweep serves both as a model-selection record and as evidence that the reported Transformer gains are not tied to a single arbitrary hyperparameter choice.

B. Transformer Implementation Details

The Transformer next-role model is implemented in PyTorch as a causal sequence model over discrete role IDs. Given a role vocabulary of size K, we reserve two additional token IDs for `[PAD]` and `[BOS]`, yielding a total token vocabulary of size $K + 2$. During training, each trace is converted into prefix–target pairs by prepending `[BOS]` and predicting each next role from its observed prefix.

Batch construction uses variable-length prefixes clipped to at most the selected context length `ctx`. Within each mini-batch, the effective sequence length is the minimum of `ctx` and the longest prefix in that batch. Prefixes longer than this limit are truncated to the most recent context, while shorter prefixes are right-padded with `[PAD]`. The attention mask marks valid tokens and is used both as a padding mask and to recover the last non-padding hidden state for prediction.

The model is implemented as a stack of Transformer encoder blocks equipped with a causal triangular mask, so that token representations depend only on the observed prefix despite using the encoder-layer primitive. The final hidden state at the last valid prefix position is passed through a linear output layer to obtain logits over the next-role vocabulary.

Training uses AdamW optimization, gradient clipping with maximum norm 1.0, shuffled mini-batches of size 256, and a fixed random seed of 42. The script allows up to 200 training epochs and selects the best checkpoint using early stopping on validation Recall@10 computed on the trigram-subset, with patience 8 and minimum improvement threshold 10^{-4}. The training, validation, and test partitions are all constructed at the application level.

The implementation also evaluates two edge sets for next-step prediction: *all edges*, which begin from the first prediction after `[BOS]`, and the *trigram-subset*, which includes only transitions with at least two preceding real roles in the observed sequence. The latter is used for checkpoint selection and for comparisons intended to align with higher-order context baselines.

The final checkpoint used in the main experiments corresponds to context length 16, learning rate 3×10^{-5}, dropout 0.3, weight decay 0.1, batch size 256, and validation ratio 0.2.

C. Additional Evaluation Details

We also evaluated a 4-gram baseline as a diagnostic of whether increasing Markov order improves atypical-transition ranking under the same fixed negative-sampling protocol. The 4-gram model did not outperform the trigram baseline (AUROC $= 0.752$, AP $= 0.309$), consistent with increased sparsity at higher orders.

For clarity, the atypical-transition results reported in the study are based on a fixed sampled discrimination setting rather than a fully natural anomaly benchmark. Accordingly, the reported AUROC, AP, and precision–recall values should be interpreted as measuring relative ranking quality under this controlled protocol.

D. Additional Low-Granularity Results for $K = 10$ and $K = 15$

To assess whether the downstream granularity sweep should extend below $K = 20$, we additionally evaluated $K = 10$ and $K = 15$. Table XII reports the corresponding next-step prediction results. These smaller values yield higher raw Recall@k, but correspond to substantially coarser role vocabularies, as reflected by their much larger median cluster sizes in Table I. For these low-granularity Transformer runs, checkpoint selection used validation Recall@5 rather than Recall@10, since Recall@10 becomes trivial at $K = 10$.

E. Multi-seed Transformer Results

Table XIII reports the seed-specific Transformer results underlying the summary in Table IV. These runs use the same $K = 40$ setting, data split, and model recipe as the main experiment, differing only in random seed. The low variation across seeds supports the stability of the reported Transformer gains.

F. Sensitivity to the Number of Sampled Negatives

We evaluated Transformer-based atypical-transition detection under four negative-sampling settings, $N_{\text{neg}} \in \{5, 10, 20, 50\}$. As shown in Table XIV, AUROC remains nearly constant across all settings (0.8317–0.8328), indicating stable relative ranking quality. In contrast, Average Precision decreases from 0.5626 at $N_{\text{neg}} = 5$ to 0.1654 at $N_{\text{neg}} = 50$, and recall at precision ≥ 0.8 decreases from 0.1720 to 0.0012. This behavior is expected because increasing N_{neg} makes the sampled evaluation more imbalanced and yields increasingly conservative high-precision thresholds. Table XIV additionally reports the selected decision threshold and the achieved precision for each setting.

TABLE XI. TOP-PERFORMING CONFIGURATIONS FROM TRANSFORMER SWEEP FOR NEXT-STEP PREDICTION AT $K = 40$

ctx	lr	dropout	val ratio	Best val. score
16	3×10^{-5}	0.3	0.2	**0.7187**
16	3×10^{-5}	0.3	0.1	0.7179
16	3×10^{-5}	0.2	0.2	0.7159
16	3×10^{-4}	0.3	0.2	0.7157
16	3×10^{-5}	0.1	0.2	0.7146
8	3×10^{-5}	0.3	0.2	0.7144
16	10^{-4}	0.3	0.2	0.7144
32	3×10^{-5}	0.3	0.2	0.7141

Top-performing configurations from the 120-run Transformer sweep for next-step prediction at $K = 40$ under the cross-application protocol. All runs used label smoothing 0.0, 200 training epochs, and patience $= 8$. The first row is the checkpoint used in the main experiments. For brevity, only the strongest configurations are shown.

TABLE XII. LOW-GRANULARITY NEXT-STEP PREDICTION RESULTS FOR $K = 10$ AND $K = 15$

K	Model	R@1	R@3	R@5	R@10
10	Unigram	0.1935	0.4698	0.6788	1.0000
10	Bigram	0.2472	0.5455	0.7780	1.0000
10	Trigram	0.3747	0.6476	0.8299	1.0000
10	Transformer	**0.4022**	**0.6976**	**0.8650**	**1.0000**
15	Unigram	0.1237	0.3672	0.5451	0.8835
15	Bigram	0.1732	0.4283	0.6196	0.9189
15	Trigram	0.3207	0.5519	0.7125	0.9305
15	Transformer	**0.3516**	**0.6193**	**0.7745**	**0.9440**

Additional low-granularity next-step prediction results used to assess whether the downstream granularity sweep should extend below $K = 20$. As expected, smaller values yield higher raw Recall@k, but correspond to substantially coarser role vocabularies. For the Transformer runs at $K = 10$ and $K = 15$, checkpoint selection used validation Recall@5 rather than Recall@10, since Recall@10 becomes trivial at $K = 10$.

TABLE XIII. SEED-SPECIFIC TRANSFORMER NEXT-STEP PREDICTION RESULTS FOR $K = 40$

Seed	R@1	R@3	R@5	R@10	R@20
42	0.2598	0.4579	0.5644	0.7268	0.8952
43	0.2729	0.4741	0.5928	0.7514	0.9054
44	0.2686	0.4754	0.5880	0.7527	0.9089
Mean $\pm$ Std	0.2671 ± 0.0066	0.4691 ± 0.0097	0.5817 ± 0.0152	0.7436 ± 0.0146	0.9032 ± 0.0071

TABLE XIV. SENSITIVITY OF TRANSFORMER ATYPICAL-TRANSITION METRICS TO THE NUMBER OF SAMPLED NEGATIVES N_{neg} ($K = 40$)

N_{neg}	AUROC ↑	AP ↑	Recall@Precision ≥ 0.8 ↑	Threshold	Achieved Precision
5	0.8317	0.5626	0.1720	0.3053	0.8003
10	0.8323	0.4231	0.0568	0.5935	0.8000
20	0.8319	0.2957	0.0102	0.8060	0.8158
50	0.8328	0.1654	0.0012	0.8986	0.8750

G. Full-Vocabulary Re-Evaluation

The full-vocabulary result in Table XV shows that the sampled-negative protocol does not materially inflate AUROC, but it does make the anomaly task appear less severe in terms of Average Precision and high-precision recall. This supports the interpretation that sampled negatives are acceptable for comparing relative ranking quality across models, while still understating the difficulty of precision-constrained screening under exhaustive candidate evaluation.

H. Full Leave-One-Category-Out Breakdown

Table XVI reports the full per-category leave-one-category-out results for the Transformer-based model at $K = 40$. The breakdown shows substantial heterogeneity across categories. The strongest held-out categories by Transformer R@10 are *Education* (0.7666), *Weather* (0.7455), and *Beauty* (0.7452), whereas the weakest are *Sports* (0.5446) and *Food & Drink* (0.5843). This supports the interpretation that the macro-average reported in the main text reflects strong average transfer, but not uniform difficulty across all UI domains.

I. Qualitative Audit of Representative Screen Roles

To assess whether the learned role abstraction at $K = 40$ corresponds to semantically coherent UI states rather than arbitrary visual partitions, we qualitatively inspected representative screens from selected high-frequency clusters. For each cluster, screens were ranked by proximity to the cluster centroid in the fused embedding space, and the nearest examples were retained while applying simple diversity constraints to reduce near-duplicate screens from the same app or trace.

Fig. 7–10 show representative examples for 12 selected clusters. These examples are intended as qualitative support

TABLE XV. COMPARISON BETWEEN THE PRIMARY SAMPLED-NEGATIVE ANOMALY PROTOCOL AND A FULL-VOCABULARY RE-EVALUATION FOR THE TRANSFORMER AT $K = 40$. BOTH SETTINGS USE THE SAME 6,058 MATCHED TEST TRANSITIONS

Evaluation setting	AUROC ↑	AP ↑	Recall@Precision ≥ 0.8 ↑
Sampled negatives ($N_{\text{neg}} = 10$)	0.8323	0.4231	0.0568
Full vocabulary (39 negatives/context)	0.8320	0.1943	0.0028

TABLE XVI. FULL LEAVE-ONE-CATEGORY-OUT BREAKDOWN FOR THE TRANSFORMER AT $K = 40$. RESULTS ARE REPORTED PER HELD-OUT CATEGORY ON THE TRIGRAM-SUBSET

Category	R@1	R@3	R@5	R@10
Education	0.2593	0.4856	0.6143	0.7666
Weather	0.1506	0.3933	0.5576	0.7455
Beauty	0.2258	0.3968	0.5290	0.7452
Dating	0.2272	0.4240	0.5786	0.7305
Video Players & Editors	0.2017	0.4177	0.5308	0.7282
Comics	0.2003	0.4035	0.5512	0.7281
House & Home	0.2778	0.4709	0.5450	0.7275
Entertainment	0.2176	0.4078	0.5285	0.7255
Auto & Vehicles	0.2320	0.4394	0.5524	0.7248
Art & Design	0.2238	0.4476	0.5429	0.7190
Business	0.2206	0.4428	0.5492	0.7162
Parenting	0.2186	0.4269	0.5439	0.7126
Social	0.2158	0.4200	0.5406	0.7119
Finance	0.2336	0.4252	0.5245	0.7023
Communication	0.1822	0.3779	0.5187	0.7006
Medical	0.2312	0.4363	0.5462	0.6946
Books & Reference	0.1973	0.3867	0.5050	0.6938
Lifestyle	0.1955	0.3849	0.5045	0.6905
Health & Fitness	0.2337	0.4376	0.5290	0.6861
Shopping	0.1952	0.3921	0.5064	0.6745
Maps & Navigation	0.2241	0.4212	0.5140	0.6742
Music & Audio	0.2406	0.4230	0.5235	0.6682
News & Magazines	0.2018	0.3691	0.4877	0.6587
Travel & Local	0.1834	0.3846	0.5022	0.6575
Food & Drink	0.1776	0.3582	0.4359	0.5843
Sports	0.1392	0.2948	0.3835	0.5446

for the interpretability of the learned role abstraction, complementing the internal clustering diagnostics reported in the main study. They should not be interpreted as a substitute for a formal human-annotation study.

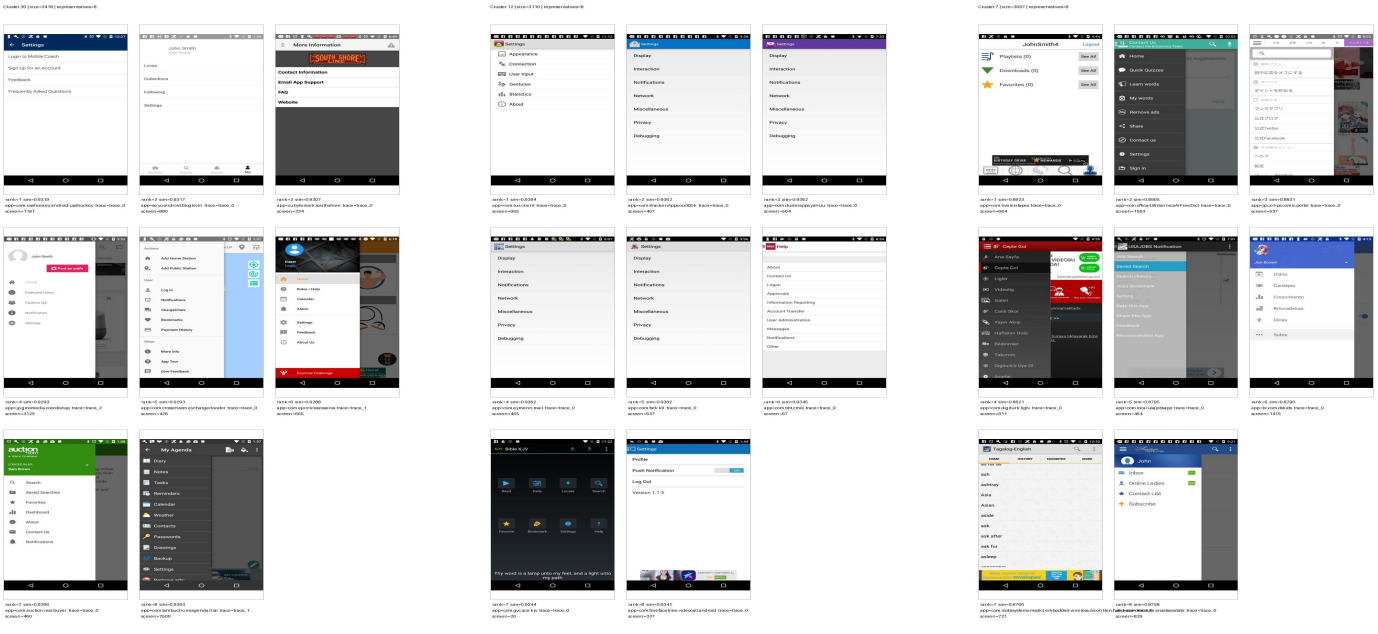

Fig. 7. Qualitative audit of representative learned screen roles at $K = 40$ (Part I).

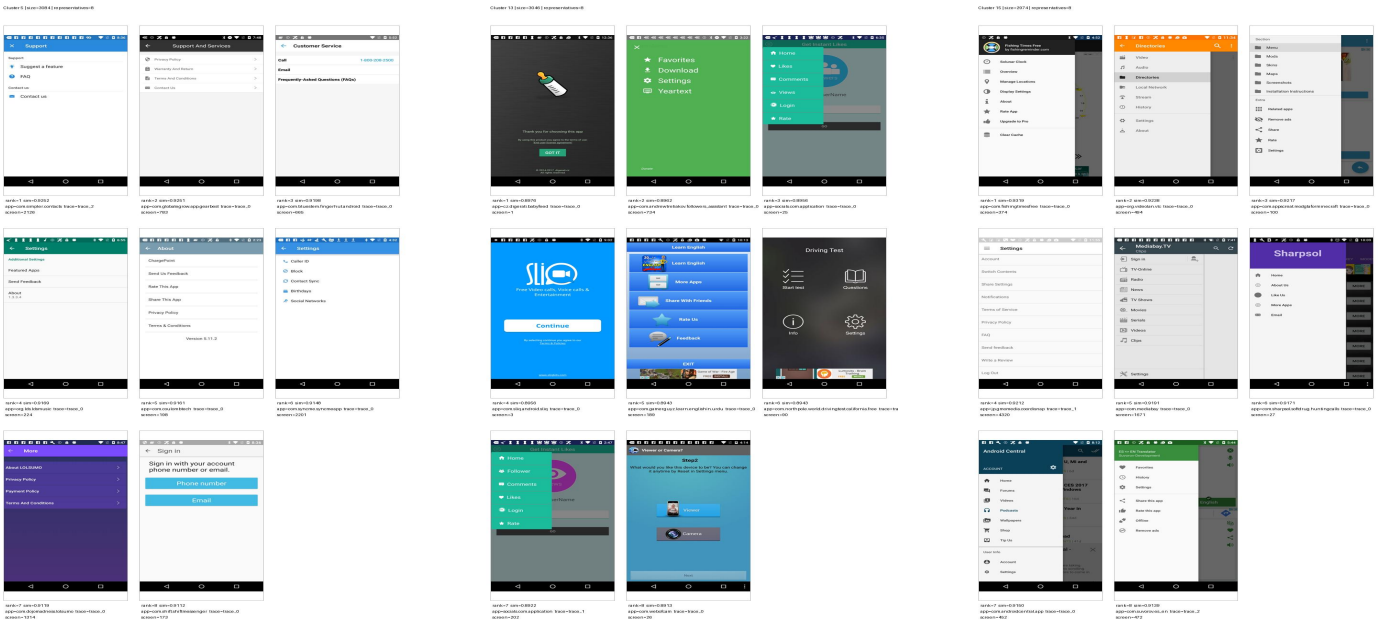

Fig. 8. Qualitative audit of representative learned screen roles at $K = 40$ (Part II).

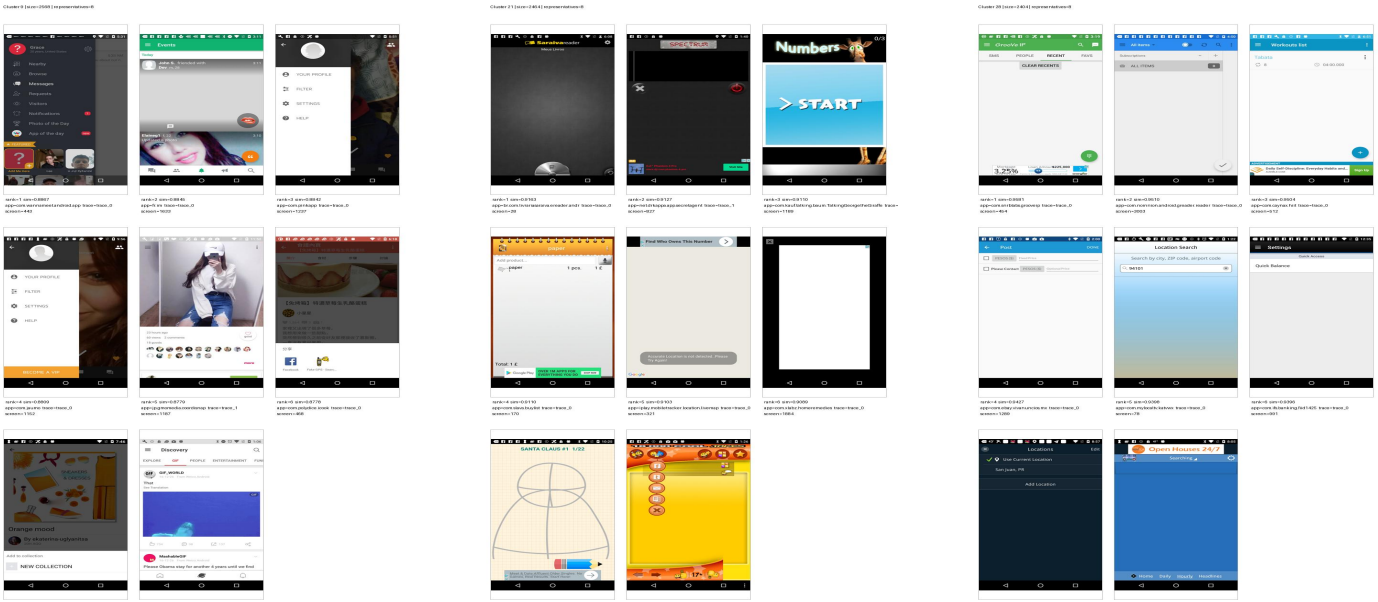

Fig. 9. Qualitative audit of representative learned screen roles at $K = 40$ (Part III).

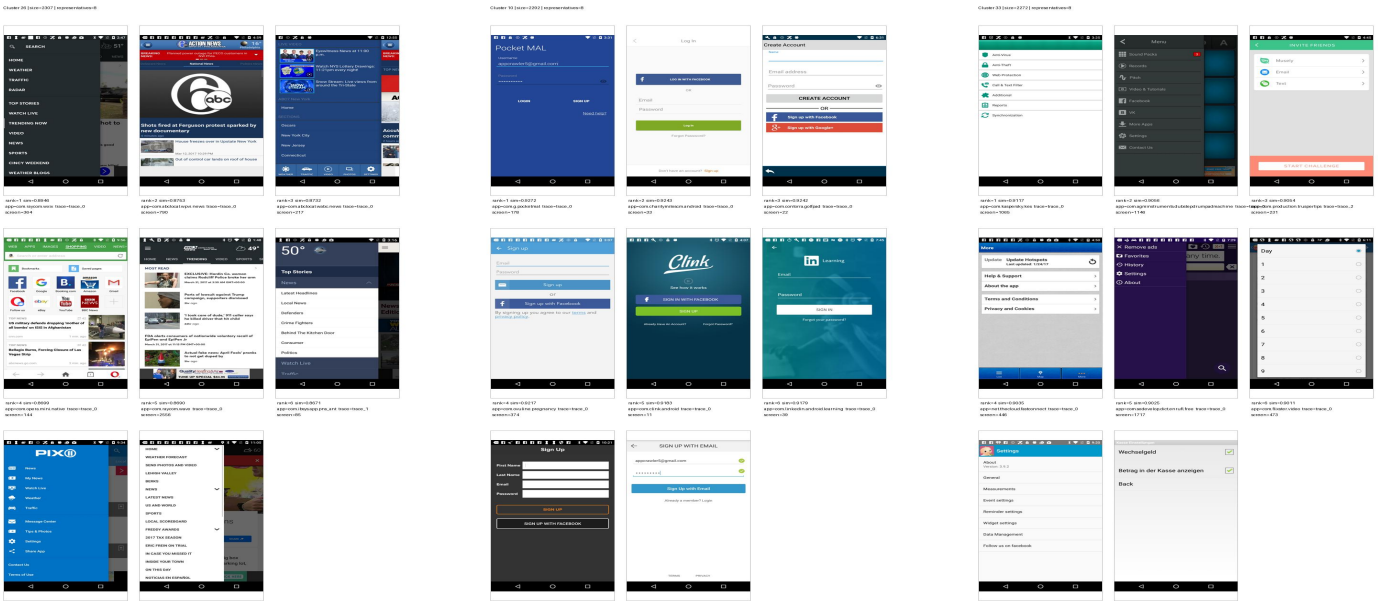

Fig. 10. Qualitative audit of representative learned screen roles at $K = 40$ (Part IV).