

IntHQ: Task-Interactive Hierarchical Query on Dual-Stream Representations for Generative Recommendation

Junjie Sun, Longfei Xu*, Huimin Yan, Wei Luo, Kaikui Liu, Xiangxiang Chu
DreamX, Alibaba Group

Abstract

Multi-task learning over heterogeneous data is fundamental to modern recommendation, while generative models are emerging as the backbone of next-generation recommenders. However, the integration of multi-task learning into the generative paradigm remains largely unexplored.

Existing multi-task recommenders, in both discriminative and generative paradigms, extract task-relevant features from a single task-agnostic representation and wire tasks into a predefined conversion funnel. We show that this scheme is inherently prone to a threefold collapse. **Source collapse**, where task-specific signals are injected late and diluted in the shared latent space. **Relational collapse**, where task dependencies are either implicitly absorbed by the backbone or statically fixed by predefined funnels. **Hierarchical collapse**, where tasks depend on features at different scales and shift across training stages.

We propose **IntHQ**, a multi-task generative recommender with three components, each alleviating one collapse. **Dual-Stream Decoupling (DSD)** injects task identity into computation stream early and separates the shared context stream from the task-specific stream, alleviating signal dilution. **Task-Interactive Modeling (TIM)** replaces the predefined funnel with explicit cross-task interaction, letting each task condition on the realized outcomes of its predecessors with learned, input-adaptive strength. **Hierarchical Querying (HQ)** lets each task gather multi-scale information across different layers at different training stages.

In offline evaluations, **IntHQ** consistently outperforms competitive encoder backbones under four representative task-head configurations. Deployed in production on Amap, serving hundreds of millions of users for travel recommendation, **IntHQ** yields a 1.60% relative UVCTR lift.

1 Introduction

Modern industrial recommender systems increasingly rely on multi-task learning (MTL) to jointly optimize multiple business objectives (Caruana, 1997; Wang et al., 2023; Zhang et al., 2025). By sharing information across tasks, MTL reduces serving cost and improves robustness under bias, sparsity, and cold-start settings (Zhang et al., 2025). Existing work mainly follows three directions. The first focuses on allocating capacity across tasks, evolving from shared-bottom (Caruana, 1997) to MMoE (Ma et al., 2018a) and PLE (Tang et al., 2020) to mitigate negative transfer. The second investigates the relation among tasks, such as ESMM (Ma et al., 2018b) and AITM (Xi et al., 2021), which leverage dependencies among user actions to reduce sparsity and selection bias. The third operates at the optimization level. GradNorm (Chen et al., 2018) rebalances per-task loss scales, PCGrad (Yu et al., 2020) projects away the conflicting component between task gradients. CAGrad (Liu et al., 2021) and Nash-MTL (Navon et al., 2022) solve for an update direction that every task

*Corresponding author.

accepts. These methods (Tang et al., 2020; Sheng et al., 2021; Wang et al., 2025; Ma et al., 2018a;b) are developed for discriminative, feature-engineering-heavy ranking stages within a multi-stage cascade, and do not transfer naturally to a unified sequence formulation.

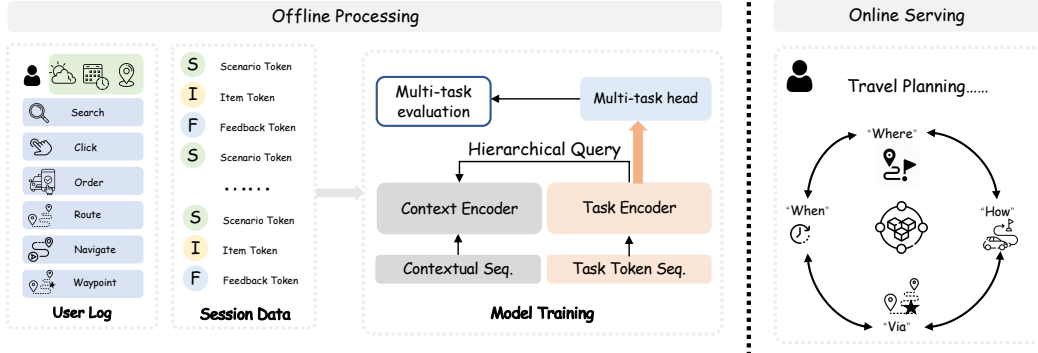


Figure 1: Overview of **IntHQ**. Left: offline processing turns user logs into session-structured data with scenario/item/feedback (SIF) tokens. The context stream and the task-token stream are decoupled. Task tokens interact within task encoder and hierarchically query the context encoder. Right: online serving jointly produces the *when*, *where*, *how*, and *via* decisions for travel recommendation, varies with time and scenarios.

Inspired by Large Language Models (LLMs), the community has recently moved toward Generative Recommendation (GR) (Hou et al., 2026; Zhai et al., 2024; Deng et al., 2025; Rajput et al., 2023; Zhang et al., 2026). Tailored for high-cardinality, non-stationary streaming data (Zhai et al., 2024), GR leverages Transformer-based sequence models to reformulate recommendation as a direct sequence generation task. By unifying historical behaviors, feedback, context, and candidates into a single tokenized sequence, GR collapses the fragmented cascade into one backbone and enables end-to-end model scaling (Zhai et al., 2024; Yan et al., 2026; 2025; Shi et al., 2024; 2026).

Although GR is becoming the backbone of a new generation of recommender systems, how to meet the multi-objective demands inherent to industrial settings remains an open problem. GR is trained under a single-objective, next-item prediction paradigm (Rajput et al., 2023; Zhai et al., 2024; Deng et al., 2025) and does not natively support the joint optimization of multiple heterogeneous objectives. To bridge this mismatch, existing multi-task generative recommenders fuse task modeling into the shared sequence representation, either conditioning tasks at output-side heads (Yan et al., 2026) or injecting task tokens that are encoded jointly with behavior tokens (Tang et al., 2026; Liang et al., 2026). Either way, task signals become entangled with the shared representation. Multi-task learning with the generative paradigm still faces three collapses.

- **Source collapse.** Task-specific signals are injected late and diluted in the shared space. Context tokens and task tokens are functionally heterogeneous. Encoded as a continuous stream, context tokens must characterize user behavior across diverse scenarios, yielding a high-dimensional representation coupled with user intention. Task tokens, in contrast, are grounded on task-specific semantic manifolds (Bengio et al., 2013; Vafidis et al., 2025). Task tokens require distinct, task-oriented emphases.
- **Relational collapse.** Task dependencies are either implicitly absorbed by the backbone or statically fixed by predefined funnels (Ma et al., 2018b; Xi et al., 2021). Interactions entangled inside the shared stream can only be curbed by cross-task gradient isolation that discards useful cross-task signal (Tang et al., 2026). Yet some multi-task dependencies are complex and cannot be specified a priori (Yan et al., 2026).

- **Hierarchical collapse.** Different layers of a deep backbone encode context at different granularities. From fine-grained local patterns to abstract long-range intent, heterogeneous tasks rely on different depths at different training stages (Sun et al., 2020). Taking only the last layer or a fixed per-layer scalar gate (Yan et al., 2026) flattens this hierarchy into one representation shared by all tasks.

The three collapses above take a concrete form in map and travel services. As a Location-Based Service (LBS) provider, we resolve four coupled decisions within a single session, namely a user’s departure time, destination, travel mode, and waypoint, which we refer to as *when*, *where*, *how*, and *via*. Figure 1 illustrates the deployed service.

Within a single session, the four tasks draw on different evidence. *Where* and *when* are settled by a coarse periodicity of the long-term history. *Via* turns on one transient need inside the current session, such as refueling. *How* responds to the real-time condition of the moment. A single task-agnostic embedding keeps the dominant statistic and drops the minority evidence, therefore creates a bottleneck. Nothing placed downstream can recover the discarded information. **Task signals therefore have to be injected early.** Meanwhile, the context computation compresses sequences into what persists, whereas task computation is discriminative in current situation. The two roles have different optima for the same parameters. A shared block can only settle between them. **The task and context computation therefore have to live in a decoupled parameter space.** Moreover, the tasks induced by user behaviors are tightly coupled rather than independent across scenarios, as shown in bottom right of Figure 1. The destination in commuting scenario shapes travel time and mode, but an uncertain destination on a trip loosens this dependence, so no fixed order holds across scenarios. **Dependencies that shift with real-world constraints call for an adaptive task-interaction framework.** Meanwhile, **the tasks differ substantially in vocabulary size and in the sparsity of informative behavioral signals**, so it is beneficial to route representations at different scales adaptively.

Motivated by these observations, we propose Task-Interactive Hierarchical Query (IntHQ). The main contributions of this work are summarized as follows:

- **Novel Multi-task Generative Recommendation Paradigm.** We identify three collapses in multi-task generative recommendations and propose IntHQ with three components that address the three collapses respectively. **Dual-Stream Decoupling.** We attach four constructed task tokens at each position of the behavior sequence to form an independent task-token sequence, with separated parameters decoupled from the context computation stream. **Task-Interactive Modeling.** Within the task-token stream, we introduce an attention-based, explicit, and adaptive interaction among tasks. **Hierarchical Querying.** Task tokens act as queries while the context-stream representations serve as keys and values, aggregating informative signals across different layers.
- **Comprehensive Experimental Validation.** We conduct extensive experiments on a real-world industrial dataset IntTravel (Yan et al., 2026). IntHQ provides highly informative task signals that consistently benefit all task heads, and outperforms the latest baselines. Ablation study also validates the contribution of each component.
- **Deployment and evaluation.** We deploy IntHQ in production at Amap. IntHQ currently serves hundreds of millions of users at tens of thousands of Query Per Second (QPS) over tens of millions of POIs, and powers key products including app launch map, terminal recommendation, travel-mode recommendation, and along-the-way recommendation. Extensive online A/B tests show consistent gains, delivering a 1.60% relative UVCTR (CTR on unique user views) lift while keeping average inference latency within 40 milliseconds.

2 Preliminaries

In this section, we formalize multi-task learning in travel recommendation. Building on this formulation, we theoretically and empirically expose a threefold collapse inherent in conventional architectures, which inspires the design principles of our approach.

2.1 Problem Formulation

Given all user interactions set $\mathcal{A}$, consists of the context sequence summarizing the user profile, spatio-temporal state, and history, the system jointly resolves a travel intention that decomposes into K coupled sub-decisions indexed by $k \in \mathcal{K}$. We instantiate $\mathcal{K} = \{\text{when, where, how, via}\}$. The ground-truth labels are defined as $\mathbf{y} = [y_k \mid k \in \mathcal{K}]$, each y_k valued in a candidate set $\mathcal{I}_k$ (possibly huge, e.g. POIs). Training pairs in dataset $\mathcal{D}$ are $(\mathcal{A}, \mathbf{y}) \sim \mathcal{D}$.

User behavior is logged by session. Interaction set $\mathcal{A}$ is an ordered sequence assembled from three token types.

$$\mathcal{A}_u = ((S_t, I_t, F_t))_{t=1}^{T_u}, \quad S_t \in \mathcal{S}, I_t \in \mathcal{I}, F_t \in \mathcal{F}, \quad (1)$$

where each step t contributes three context token types. Scenario token S_t encodes spatio-temporal state of the t -th action. Item token I_t denotes the POI interacted with at step t and its attributes. Feedback token F_t signals the user's interaction type, such as clicking or searching. In addition, a set of user profile tokens $\mathcal{U}_u = (U_1, \dots, U_m)$, encoding long-term static attributes (e.g., demographics), is prepended to the sequence, i.e., $\mathcal{A}'_u = (\mathcal{U}_u, (S_t, I_t, F_t)_{t=1}^{T_u}) \in \mathcal{A}$, serving as a global context visible to all subsequent tokens. To decouple task tokens $q_k^t \in \mathcal{Q}$ and context tokens, we define task token sequence and attach labels to related tasks as:

$$\mathcal{Q}_u = ((q_{\text{when}}^t, q_{\text{where}}^t, q_{\text{how}}^t, q_{\text{via}}^t))_{t=1}^{T_u}, \quad q_k^t \in \mathcal{Q}_k, \quad k \in \mathcal{K}, \quad (2)$$

A model with parameters Θ encodes the request into a representation $z = \text{Enc}_\Theta(\mathcal{A})$ and is trained by maximum likelihood of the joint target.

$$\hat{\Theta} = \arg \max_{\Theta} \sum_{(\mathcal{A}, \mathbf{y}) \in \mathcal{D}} \log P([y_k \mid k \in \mathcal{K}] \mid \mathcal{A}; \Theta), \quad (3)$$

2.2 Pilot Study

We score a model by the joint log loss of the outcome tuple, whose Bayes optimum is $R^* = H(\mathbf{y} \mid \mathcal{A})$, where H denotes conditional entropy. Conventional multi-task recommenders, in both the discriminative and the generative paradigm, share one pattern. They compress $\mathcal{A}$ into a single task-agnostic code $z = \text{Enc}(\mathcal{A})$ and score the tuple with heads that are conditionally independent given z , so the predicted law is $\prod_k q_k(y_k \mid z)$. We call this *Task-Agnostic Encoding* (TAE), and its optimal risk is $R_{\text{TAE}}(z) = \sum_k H(y_k \mid z)$. Theorem 1 splits the excess risk of TAE into two non-negative terms, which we name the **source** and the **relation** collapse. All proofs and analysis on **source** and **relation** collapse are in Appendix A. The **hierarchy** collapse is established empirically in §4.1.

Theorem 1 (Collapse decomposition). *Assume every $\mathcal{I}_k$ is finite, $z = \text{Enc}(\mathcal{A})$ is a measurable deterministic map, and the heads range over all conditional laws with no parameter shared across them. Then*

$$R_{\text{TAE}}(z) - R^* = \underbrace{\sum_k I(\mathcal{A}; y_k \mid z)}_{\Delta_{\text{src}}(z)} + \underbrace{\text{TC}(\mathbf{y} \mid \mathcal{A})}_{\Delta_{\text{rel}}}, \quad (4)$$

where $I(\mathcal{A}; y_k \mid z) = H(y_k \mid z) - H(y_k \mid \mathcal{A})$ is the information about y_k that the encoding discards, namely total correlation (Watanabe, 1960; Ver Steeg & Galstyan, 2014) $\text{TC}(\mathbf{y} \mid \mathcal{A}) = \sum_k H(y_k \mid \mathcal{A}) - H(\mathbf{y} \mid \mathcal{A})$. Both terms are non-negative, so $R_{\text{TAE}}(z) \geq R^* + \Delta_{\text{rel}}$ for every encoder, with equality if and only if the encoding z marginally sufficient for every task.

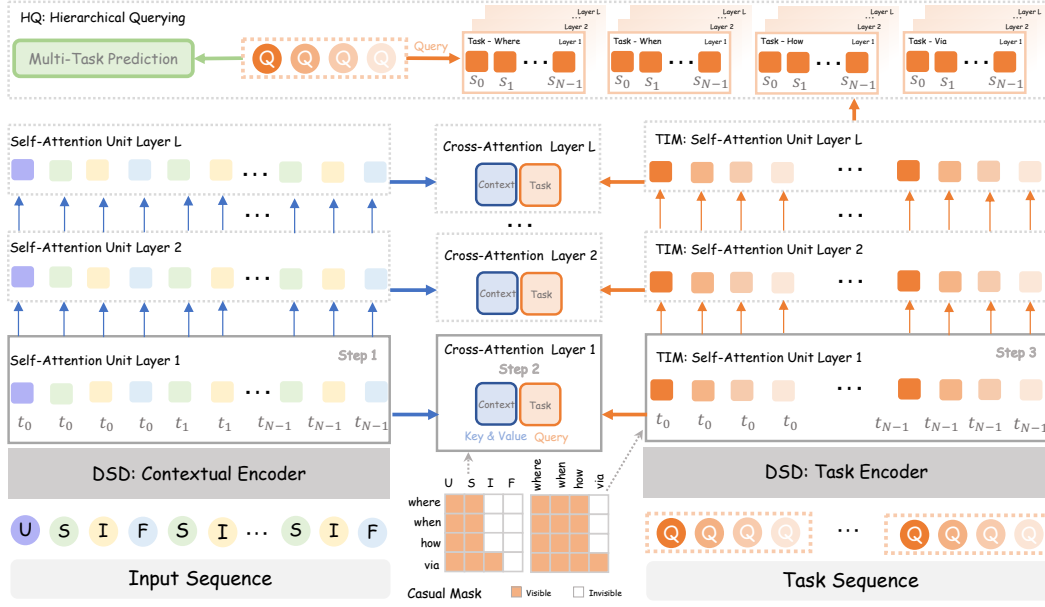


Figure 2: The detailed architecture of IntHQ. IntHQ is composed of three components, Dual-Stream Decoupling, Task-Interactive Modeling and Hierarchical Querying. The two streams are encoded separately and never concatenated. The mask between task tokens and context tokens is illustrated in the figure below, while visibility within the context tokens follows standard causal attention.

The two terms have disjoint dependence and therefore demand different structures. $\Delta_{\text{src}}(z)$ carries the encoder and vanishes once z is marginally sufficient per task, so a better encoder can shrink it. Δ_{rel} denotes the floor of relation collapse and carries no z , so it bounds every factorized read-out from below no matter how large the encoder is or how it is trained. Shrinking Δ_{src} asks for task conditioning inside the encoder, which we realize as *Dual-Stream Decoupling*. Alleviating Δ_{rel} requires decisions that condition on one another, which we realize as *Task-Interactive Modeling*.

3 Methodology

In this section, we present **IntHQ**. As illustrated in Figure 1, **IntHQ** targets multi-task travel recommendation, where predictions are grounded in real-world spatiotemporal constraints and the dependencies among tasks shift dynamically with the scenario. **IntHQ** targets three collapses found in the pilot study (§2.2) with three core components as shown in Figure 2. **Dual-Stream Decoupling (DSD)**, targeting the source collapse, separates context encoding from task encoding. **Task-Interactive Modeling (TIM)** alleviates the relation collapse by enabling explicit and direct interactions among tasks. **Hierarchical Querying (HQ)** lets each task token gather information at multiple scales across layers. In the following subsections, we describe each component in detail.

3.1 DSD: Dual-Stream Decoupling

The Dual-Stream Decoupling (DSD) module captures user intent and task-specific inclinations. DSD takes the session-structured user behavior sequence $\mathcal{A}$ as input and correspondingly constructed sequence of task tokens Q_u . Two streams are encoded and interacted in decoupled latent spaces.

3.1.1 Sequence Construction.

Raw interaction logs are heterogeneous. Each decision is made under an observable context, lands on a concrete item, and ends with a certain type of feedback. We map every logged session to three type context tokens. The scenario token S encodes the decision context, such as time, location and weather. The item token I encodes the consumed object and its attributes, e.g., POIs that users interacted. The feedback token F encodes how the user engaged with the item, e.g., click, search, or order. A small set of profile tokens U carrying long-term user attributes is appended once per sequence, serving as a global, time-invariant context.

The sequence $\mathcal{A}_u$ is organized session by session in temporal order, with all tokens of a session sharing the same timestamp, as in Eq. 1. The session is the natural unit of a decision. In travel, one trip jointly realizes *when* to depart, *where* to go, *how* to travel, and *via* which route.

In parallel to $\mathcal{A}_u$, we construct the task stream $\mathcal{Q}_u$, as in Eq. 2. Each task token q_k is a task-specific learnable parameter shared across all sequences. It serves as a query template for task k and is instantiated once per session. Task tokens carry the supervision of its task and query the context under a causal constraint that forbids label leakage.

3.1.2 Decoupled Dual-Stream Attention.

The two streams are encoded by independent attention cores with disjoint parameters. Within each layer, context core Attn_{ctx} and task core Attn_{q} encoding proceeds in two steps.

Context self-attention. The context stream applies self-attention over all context tokens.

$$\tilde{H}_{\text{ctx}}^{(\ell)} = \text{Attn}_{\text{ctx}}(H_{\text{ctx}}^{(\ell)}, H_{\text{ctx}}^{(\ell)}), \quad (5)$$

where $H_{\text{ctx}}^{(\ell)}$ denotes the context representations at layer ℓ . This step distills a task-agnostic summary of user intent from the behavioral history.

Task context cross-attention. Each task token queries the context stream.

$$\tilde{H}_{\text{q}}^{(\ell)} = \text{Attn}_{\text{q}}(H_{\text{q}}^{(\ell)}, H_{\text{ctx}}^{(\ell)}), \quad (6)$$

so that the query, key, and value projections of Attn_{q} are learned *exclusively* from task-token gradients, while the key–value context is supplied by the context stream. Crucially, the key–value set is drawn from the *input* of the current layer $H_{\text{ctx}}^{(\ell)}$, not from the self-attended output $\tilde{H}_{\text{ctx}}^{(\ell)}$, so that the two steps remain computationally independent within a layer.

Causal visibility. All attention operations in **IntHQ**, context self-attention, task–context cross-attention, and the task self-attention introduced in §3.2 obey the same visibility rule. Query at position p_q may attend to a key at position p_k if and only if $c(p_k) \leq c(p_q)$, where $c(\cdot)$ is a scalar causal index assigned during sequence construction. The index encodes two constraints simultaneously: (i) *cross-session causality*. Tokens in earlier sessions are visible to all later tokens; and (ii) *intra-session ordering*. Within a session, a token sees only the facets that causally precede it, e.g., a *via* token sees the realized *where* token, but not vice versa. Profile tokens receive the smallest causal index and are therefore visible to every query. Following prior work, we add a learnable relative-position bias and a time-delta bias to the attention module, both indexed by the same positional structure that defines the causal index $c(\cdot)$. Figure 2 visualizes the causal rule in task-context cross attention and self-attention in task tokens, while visibility within the context tokens follows standard causal attention.

3.2 TIM: Task-Interactive Modeling

Dual-Stream Decoupling equips each task token with a context-conditioned representation via cross-attention, but these representations are computed independently. Task tokens do

not yet see one another. In real-world multi-task recommendation, the dependency of tasks is usually non-trivial. Take travel recommendation as an example, the choice of destination constrains the feasible transport modes, which in turn narrows the departure time. Leaving such dependencies to be absorbed implicitly by the shared context stream is precisely the relation collapse identified in §2.2. Task-Interactive Modeling (TIM) alleviates this gap by adding a self-attention step over the task tokens.

Task self-attention. At each layer, the task stream applies self-attention using the same decoupled core Attn_q in Equation 6:

$$\hat{H}_q^{(\ell)} = \text{Attn}_q(H_q^{(\ell)}, H_q^{(\ell)}). \quad (7)$$

The task token representations are context-conditioned. Self-attention therefore operates on representations that are both *task-identified* by the learnable embedding e_k and *session-grounded* by the accumulated cross-attention history. This completes an explicit interaction among task tokens that is learned from data rather than prescribed by a predefined task structure.

Causal visibility among tasks. The self-attention obeys the same visibility rule $c(p_k) \leq c(p_q)$ introduced in §3.1. *via* token can attend to the *where*, *when* and *how* tokens in the same session, conditioning the waypoint on the already-decided destination and mode, whereas the reverse is blocked. This ordering prevents label leakage while still allowing downstream facets to exploit upstream decisions.

Inter-Layer Connections. The attention calculation in §3.1 and §3.2 are both computed from the layer input, and enter a single residual update.

$$H_q^{(\ell+1)} = H_q^{(\ell)} + \underbrace{\tilde{H}_q^{(\ell)}}_{\text{context}} + \underbrace{\hat{H}_q^{(\ell)}}_{\text{cross-task}}, \quad H_{\text{ctx}}^{(\ell+1)} = H_{\text{ctx}}^{(\ell)} + \tilde{H}_{\text{ctx}}^{(\ell)}, \quad (8)$$

3.3 HQ: Hierarchical Querying

Section 2.2 shows that the most informative depth differs across tasks and moves during training. A single final-layer read-out flattens these differences. Hierarchical Querying lets each task select the depth it needs, using only the task stream.

Per-layer collection. At every encoder layer ℓ , the task-stream output is aggregated across residual paths, normalized, and stored. §3.1 and §3.2 yield a depth-indexed bank $\{h_q^{(1)}, \dots, h_q^{(L)}\}$ of task representations. We collect from the task stream rather than the context stream because the task-token representations have already been conditioned on each task’s information needs through cross-attention. The depth aggregation in HQ therefore only needs to select which depth matters, not which information is task-relevant.

Depth attention. For task k we stack the L layer states collected above into $Z_k = [H_{q,k}^{(1)}; \dots; H_{q,k}^{(L)}] \in \mathbb{R}^{L \times d}$, and attend over its depth axis. The query is formed from the task identity, and each layer is mapped to a key and a value with a learnable depth embedding d_ℓ marking which layer it came from,

$$\begin{cases} Q_k = W_Q q_k, & K_\ell = W_K Z_k[\ell] + d_\ell, & V_\ell = W_V Z_k[\ell], \\ \alpha_\ell^{(k)} = \frac{\exp(\langle Q_k, K_\ell \rangle / \sqrt{d_a})}{\sum_{\ell'=1}^L \exp(\langle Q_k, K_{\ell'} \rangle / \sqrt{d_a})}, \\ z_k = \text{LN}\left(H_{q,k}^{(L)} + \sum_{\ell=1}^L \alpha_\ell^{(k)} V_\ell\right). \end{cases} \quad (9)$$

where q_k is the learnable embedding of task k . The score in (9) depends on the task through q_k and on the context of each layer through κ_ℓ . The selected depth varies with the task, the session, and the user, rather than being fixed by a schedule.

3.4 Training and Loss

Each task is trained by its own InfoNCE loss (Oord et al., 2018). The output of Hierarchical Querying is passed through a task-specific head to obtain the final task representation. Note that **IntHQ** is head-agnostic. We further prove out **IntHQ** paradigm outperforms different generative backbone across different task heads in Table 1. The loss maximizes the probability of the ground-truth item i^+ over $\mathcal{I}_k$. The final multi-task loss L is the sum of the per-task losses, as shown in Eq. 10. The overall training procedure is summarized in Algorithm 1.

Algorithm 1 IntHQ forward and training step

Require: context embeddings $H_{\text{ctx}}^{(0)}$, task embeddings $H_q^{(0)}$ with $H_q^{(0)}[t, k] = e_{\text{cls}} + e_k$, layers L , decoupled cores Attn_{ctx} , Attn_q , causal masks M_{cc} , M_{qc} , M_{qq}

Ensure: multi-task loss $\mathcal{L}$

- 1: **for** $\ell = 0$ **to** $L - 1$ **do** ▷ the three calls below read the same layer input and run in parallel
- 2: $\tilde{H}_{\text{ctx}} \leftarrow \text{Attn}_{\text{ctx}}(H_{\text{ctx}}^{(\ell)}, H_{\text{ctx}}^{(\ell)}; M_{\text{cc}})$ ▷ context self-attention
- 3: $\tilde{H}_q \leftarrow \text{Attn}_q(H_q^{(\ell)}, H_{\text{ctx}}^{(\ell)}; M_{\text{qc}})$ ▷ task→context cross-attention
- 4: $\hat{H}_q \leftarrow \text{Attn}_q(H_q^{(\ell)}, H_q^{(\ell)}; M_{\text{qq}})$ ▷ task self-attention
- 5: $H_{\text{ctx}}^{(\ell+1)} \leftarrow H_{\text{ctx}}^{(\ell)} + \tilde{H}_{\text{ctx}}$ ▷ residual update, task-free
- 6: $H_q^{(\ell+1)} \leftarrow H_q^{(\ell)} + \tilde{H}_q + \hat{H}_q$
- 7: $h_q^{(\ell+1)} \leftarrow \text{LN}^{(\ell+1)}(H_q^{(\ell+1)})$ ▷ keep for hierarchical querying
- 8: **end for**
- 9: $\mathcal{L} \leftarrow 0$
- 10: **for** each task $k \in \mathcal{K}$ **do**
- 11: $z_k \leftarrow \text{DepthAttn}(e_k, \{h_q^{(\ell)}[\cdot, k]\}_{\ell=1}^L)$ ▷ hierarchical querying
- 12: $o_k \leftarrow \text{Head}_k(z_k)$
- 13: $\mathcal{L} \leftarrow \mathcal{L} + \text{InfoNCE}(o_k, i^+; \mathcal{I}_k)$ ▷ Eq. 10
- 14: **end for**
- 15: update parameters by $\nabla \mathcal{L}$; **return** $\mathcal{L}$

We build the candidate set $\mathcal{I}_k$ per task. For a task with a small label space, such as *how* and *when*, $\mathcal{I}_k$ is the full class set and the loss is a standard softmax cross entropy. For a task with a very large label space, such as *where* and *via* over POIs, we replace the full softmax by a sampled softmax, so $\mathcal{I}_k$ is the ground-truth item together with a set of sampled negatives. **IntHQ**, including the two streams, the task interaction, and the hierarchical query, is trained end to end by minimizing L .

$$L = \sum_{k \in \mathcal{K}} -\frac{1}{|\mathcal{A}|} \sum_{u \in \mathcal{U}} \sum_{a \in \mathcal{A}_u} \log \frac{\exp(\hat{y}_k^{i^+})}{\sum_{i \in \mathcal{I}_k} \exp(\hat{y}_k^i)}. \quad (10)$$

4 Experiment

4.1 Diagnosing Hierarchical Collapse

To align with our online system, we train an $L=16$ layers baseline model, implemented as IntTravel (Yan et al., 2026), on production data ¹ and attach per-layer linear probing network. For each layer ℓ , a probe predicts y_k from z_ℓ . Figure 3 reports the per-layer probed accuracy over training. The most-decodable layer differs systematically across tasks. Each task’s best layer (black curve) drifts throughout training, so no fixed depth is optimal even for one task across time. Fine-grained retrieval (*Where*, *Via*) peaks at shallow layers ($\ell \approx 1-3$), whereas coarse intent (*How*, *When*) relies on *deeper*, more abstract features ($\ell \approx 5-7$).

¹The production data follows the same data format as the public benchmark (Yan et al., 2026), and differs only in scale.

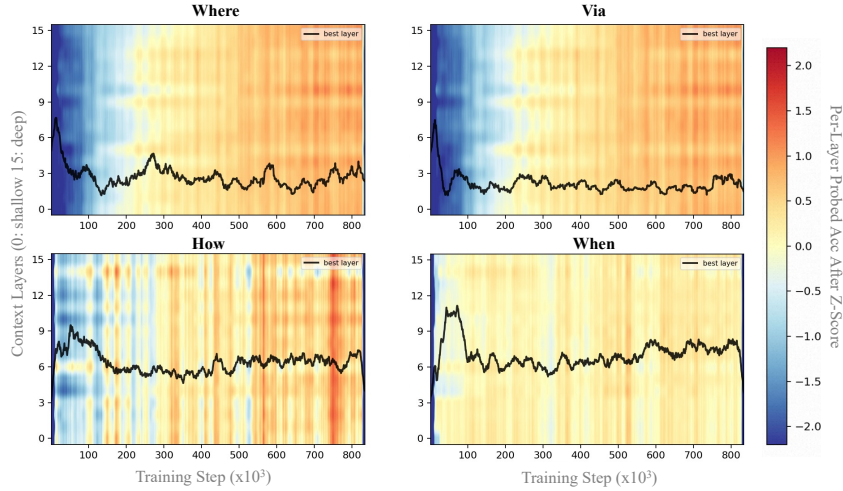


Figure 3: **Per-layer probed accuracy over training.** Color is the per-layer probed accuracy z-scored along training (warm=high, cool=low). The black curve is the smoothed best layer $\arg \max_{\ell}$. For different tasks, the reliance on features of different scales varies at different stages of training.

Intuitively, depth is governed by the range of context each task consumes. Shallow layers preserve high-resolution, immediate signals from the current session, while deeper layers progressively aggregate behaviors into long-range patterns, and tasks benefit from the two ends differently. Travel-mode prediction (*How*) profits from deep features that accumulate a user’s commuting regularities across many sessions. Destination retrieval (*Where*) hinges on the session-level scenario. The immediate spatio-temporal context exposes in shallow, high-resolution features.

4.2 Experiment Settings

4.2.1 Datasets.

We adopt IntTravel (Yan et al., 2026), a large-scale dataset built from logs of a real-world industrial map and navigation platform. It contains about 4.1 billion interactions from 163 million users over 7.3 million POIs, which is two to three orders of magnitude larger than prior benchmarks (Cheng et al., 2011; Cho et al., 2011; Yang et al., 2014; 2016; Monti et al., 2018; Yang et al., 2019), and it covers four coupled tasks: *when* to depart, *where* to go, *how* to travel, and *via* which on-the-way POI to visit.

4.2.2 Baselines.

We organize baselines along two orthogonal axes: the *sequence encoder*, which determines whether and how task signals enter the encoding computation, and the *multi-task head*, which determines how the encoded representation is shared and specialized across tasks.

Sequence encoders. We compare four encoders that form a progression of task-signal injection. **IntTravel** (Yan et al., 2026) is a HSTU-style backbone tailored to travel recommendation. It encodes the session-structured sequence without any task token, and tasks branch off only after encoding. **OneTrans** (Zhang et al., 2026) unifies sequential and non-sequential features into one token sequence with mixed parameterization. **HGenPush** (Liang et al., 2026) takes a first step toward task-aware encoding. Following decoder-only organization of user behaviors and per-branch *[cls]* tokens, task tokens are inserted into the main sequence, but they share one stream and one set of parameters with the context tokens.

Table 1: Offline performance comparison of four encoders paired with four multi-task heads on the IntTravel dataset. $\uparrow$ indicates higher is better and $\downarrow$ lower is better. Within each multi-task head, the best encoder is highlighted with a blue background and the second best is in blue (t -test, p -value < 0.01).

Encoder	Head	When		How		Where			Via		
		Acc $\uparrow$	MAE $\downarrow$	Acc $\uparrow$	BCR $\downarrow$	HR@1 $\uparrow$	HR@5 $\uparrow$	CIR $\downarrow$	HR@1 $\uparrow$	HR@5 $\uparrow$	CIR $\downarrow$
IntTravel	PLE	0.8333	8.0003	0.6779	0.0691	0.6731	0.8652	0.2474	0.6760	0.8659	0.2459
	STAR	0.8327	8.0301	0.6735	0.0708	0.6931	0.8752	0.2327	0.7032	0.8785	0.2259
	DSFNet	0.8331	8.0127	0.6740	0.0714	0.6956	0.8757	0.2305	0.7050	0.8788	0.2242
	HoME	0.8328	8.0249	0.6769	0.0702	0.6720	0.8645	0.2484	0.6778	0.8663	0.2445
OneTrans	PLE	0.8314	8.0910	0.6374	0.0942	0.6443	0.8583	0.2716	0.6765	0.8663	0.2455
	STAR	0.8310	8.1135	0.6147	0.0995	0.6750	0.8694	0.2480	0.7034	0.8772	0.2252
	DSFNet	0.8311	8.1048	0.6338	0.0863	0.6679	0.8687	0.2532	0.7034	0.8776	0.2248
	HoME	0.8315	8.0895	0.6480	0.0849	0.6474	0.8589	0.2691	0.6800	0.8672	0.2430
HGenPush	PLE	0.8310	8.1119	0.5677	0.1236	0.6556	0.8538	0.2643	0.6724	0.8639	0.2494
	STAR	0.8310	8.1114	0.5657	0.1253	0.6724	0.8603	0.2531	0.7025	0.8763	0.2270
	DSFNet	0.8310	8.1123	0.5654	0.1284	0.6672	0.8601	0.2562	0.7019	0.8765	0.2268
	HoME	0.8309	8.1116	0.5674	0.1242	0.6537	0.8528	0.2657	0.6773	0.8656	0.2452
IntHQ (Ours)	PLE	0.8348	7.9289	0.6907	0.0652	0.6842	0.8707	0.2395	0.6839	0.8706	0.2400
	STAR	0.8347	7.9366	0.6899	0.0654	0.7050	0.8819	0.2239	0.7129	0.8830	0.2188
	DSFNet	0.8347	7.9341	0.6896	0.0656	0.7050	0.8828	0.2232	0.7119	0.8841	0.2190
	HoME	0.8345	7.9433	0.6896	0.0657	0.6839	0.8712	0.2397	0.6818	0.8699	0.2424

Multi-task heads. Each encoder is paired with four representative heads that cover the mainstream design space of multi-task specialization. **PLE** (Tang et al., 2020) routes shared and task-specific experts through progressive layered extraction to mitigate negative transfer. **STAR** (Sheng et al., 2021) multiplies a shared centered network with task-specific weights in a star topology. **DSFNet** (Yu et al., 2024) is our production head with per-task private experts on top of a shared expert pool. **HoME** (Wang et al., 2025) organizes hierarchical mixture-of-experts with feature gating and self-gated residuals for grouped tasks.

4.2.3 Evaluation Metrics.

We follow the evaluation protocol of IntTravel (Yan et al., 2026), pairing each task with positive indicators and a negative indicator to capture both prediction quality and user-experience cost. For the *when* task, we report Accuracy (Acc $\uparrow$), the exact-match rate of the predicted departure time, and Mean Absolute Error (MAE $\downarrow$) as the average deviation from the ground truth. For the *how* task, we report Acc $\uparrow$ over travel modes and the Bad Case Rate (BCR $\downarrow$), defined as the top-3 miss rate, since navigation apps display only three modes on the primary screen. For the *where* and *via* tasks, we report HitRate@{1,5} $\uparrow$ over the candidate POIs and the Category Inconsistency Rate (CIR $\downarrow$), the rate at which the top-1 prediction neither matches the ground-truth POI nor shares its category.

4.2.4 Implementation Details.

All models share identical training configurations. The embedding dimension is 96, the maximum sequence length to 120, and the batch size is 64 per worker on 8 PPU GPUs. For a fair evaluation, we follow the experimental setup of IntTravel. All encoders are stacked with 3 layers. Models are trained with AdamW (learning rate 8×10^{-4} , weight decay 10^{-6}) for one epoch. The POI-retrieval tasks (*where*, *via*) are trained with sampled softmax over 64 negatives per positive: 14 drawn uniformly from the full POI vocabulary, and 50 distance-based hard negatives. The context and task streams use fully decoupled attention parameters, and the hierarchical query attention uses dimension 64 with a single head. Baseline encoders (OneTrans, HGenPush) use 4-head attention with an FFN. Baseline heads adopt the hyper-parameters reported in their original papers, adapted to the shared embedding dimension. All encoder-head combinations are trained with the same data order and random seeds for comparability.

Table 2: Ablation studies of **IntHQ** on the four tasks. We report Acc for *when/how* and HR@1 for *where/via*.

Task (Metric)	IntHQ (full)	w/o DSD	w/o TIM	w/o HQ
<i>When</i> (Acc $\uparrow$)	0.8347	0.8329	0.8347	0.8330
<i>Where</i> (HR@1 $\uparrow$)	0.7050	0.6893	0.7049	0.7004
<i>How</i> (Acc $\uparrow$)	0.6896	0.6728	0.6867	0.6757
<i>Via</i> (HR@1 $\uparrow$)	0.7119	0.6991	0.7102	0.7030

4.3 Overall Performance Comparison

We evaluate **IntHQ** and all baselines on the industrial-scale IntTravel dataset, with the full encoder-head grid reported in Table 1. Under every head, **IntHQ** outperforms all encoder baselines on all ten metrics, improving positive and negative indicators simultaneously. The gains are a uniform expansion of the quality frontier rather than a trade-off among tasks. As for the encoder axis, IntTravel strengthens the backbone but carries no task token and no task interaction. Tasks branch off only after a task-agnostic encoding, leaving both collapses intact. OneTrans unifies all features into a single stream, so task signals never enter the encoder, and its single-depth read-out serves every task from the same layer. The two ends of the depth spectrum are then in direct competition, and the loss is dominated by the large-vocabulary tasks, so the coarse-intent tasks *How* and *When*, which rely on abstract long-range features, degrade the most. HGenPush does insert task tokens, but share one stream, as source collapse predicts, the two roles dilute each other, and it performs worst precisely on the same task (*How*). **IntHQ** completes the progression. Through early task injection in a decoupled stream, explicit task interaction, and hierarchical read-out, **IntHQ** achieves the largest margins exactly where the baselines fail. Meanwhile, within **IntHQ** the performance spread across the four heads narrows markedly, encoder-side task conditioning absorbs most of the specialization work, confirming that the advantage stems from the structure itself rather than any particular multi-task head. Notably, these improvements come at an acceptable computational cost. We detail the comparison in Appendix B, taking STAR as a representative task head for illustration.

4.4 Ablation Study

Table 2 ablates the three components of **IntHQ**, each of which proves necessary, and the pattern of degradation aligns with the collapse each component targets. Replacing **DSD** as the shared encoder causes significant and uniform drops across all four tasks, as expected for source collapse, whose dilution effect is global rather than task-specific. Removing **TIM** leaves the driver tasks (*When*, *Where*) nearly untouched but visibly hurts *How* and *Via*, the facets whose decisions are conditioned on other tasks. This asymmetry is direct evidence that the gains of explicit task interaction come from modeling inter-task coupling. Removing **HQ** produces degradation on four tasks, confirming that a single read-out depth cannot serve heterogeneous granularity.

4.5 Scaling Law

A central promise of generative recommendation is that performance scales with compute. We verify that **IntHQ** inherits this property by sweeping the encoder depth $L \in \{2, 4, 8, 16, 32\}$ and width $D \in \{24, 48, 96, 192, 384\}$ with all other hyper-parameters fixed, training each variant on the same data stream. Figure 4 shows related metrics across four tasks. Deeper and wider encoders present better performance as expected. The dual-stream structure scales without modification. Adding model capacity enlarges both the context stream’s capacity for long-range patterns and the bank of representations that Hierarchical Querying can draw from.

4.6 Online Deployment

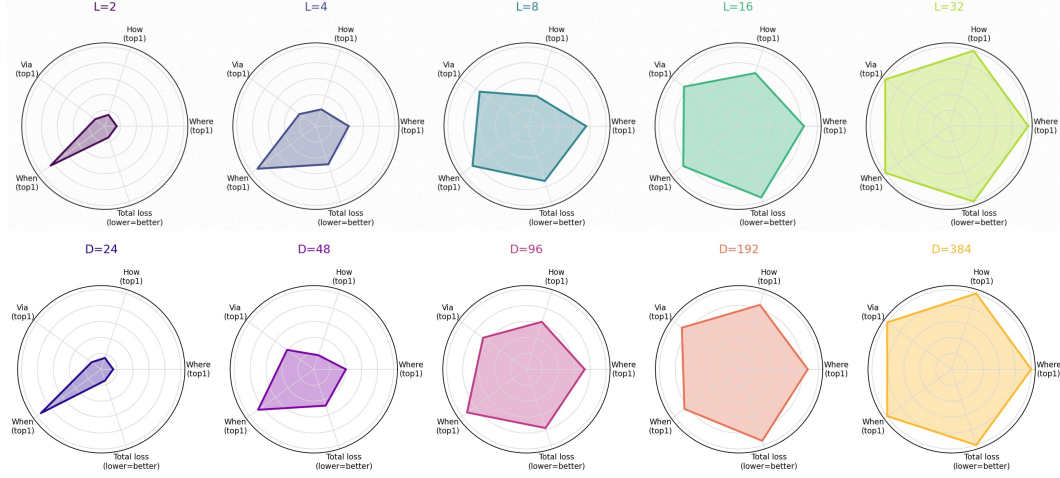


Figure 4: Scaling with encoder depth and embedding dimension. Total loss, accuracy and hit rate of **IntHQ** with $L \in \{2, 4, 8, 16, 32\}$ layers and $D \in \{24, 48, 96, 192, 384\}$ dimensions. Deeper and wider encoders converge to better performance, exhibiting clear depth-wise and width-wise scaling trends.

4.6.1 Deployment Pipeline.

Serving **IntHQ** in production must reconcile two competing demands. The model has to track travel behavior that shifts daily while every request must be answered within a tight latency budget. Figure 5 shows how we resolve them by moving all heavy, user-level computation off the request path. Sequence assembly is precomputed rather than performed online. An offline job materializes each user’s session structured history from raw behavior logs into a key-value store. At request time the model service retrieves this prefix by user ID. The assembled sequence is scored on GPU computation nodes. For *where* and *via* the resulting task representations are matched against the POI index by approximate nearest neighbor search. Response generated by **IntHQ** service and user feedback are written back to the log store, from which the next training round is assembled and the refreshed model is republished to the serving engines. Overall, **IntHQ** handles a sequence length of 300, covering 100 most recent behavioral interactions, under a peak concurrency of 30k QPS (query per second). Deployed on a cluster of 150 NVIDIA PPU GPUs (96 GB memory), the average response time (RT) is 40 ms and the 99% RT is 100 ms.

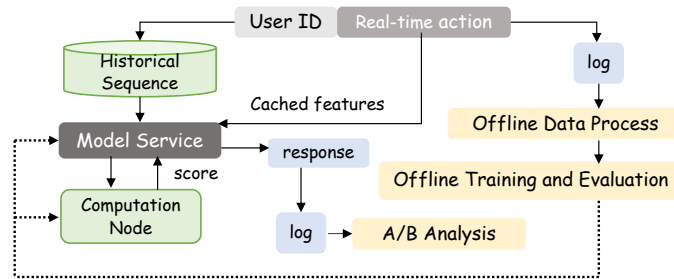


Figure 5: Online deployment architecture of **IntHQ** on Amap. Solid arrows denote the real-time serving flow. Dashed arrows denote the offline training loop that redeploys updated models to the serving engines.

4.6.2 Online A/B Results.

We evaluated **IntHQ** against the online baseline in the travel recommendation scenario of Amap, with each bucket receiving 20% of live traffic over 7 days. **IntHQ** improved UVCTR by 1.60% relative. The gain held consistently across daily slices. Handling 30k QPS on a cluster of PPU GPUs, the average response time (RT) of **IntHQ** is 40 ms. **IntHQ** has since been fully rolled out and now serves all traffic in this scenario.

5 Related Work

5.1 Classical MTL

Early shared-bottom networks and Cross-Stitch (Misra et al., 2016) share a single trunk and branch into per-task towers. Expert-based designs refine this sharing. MMoE (Ma et al., 2018a) routes soft-selected experts per task, and PLE (Tang et al., 2020) explicitly separates shared and task-specific experts through progressive layered extraction to curb negative transfer. A parallel line targets multi-domain/multi-scenario settings. STAR (Sheng et al., 2021) multiplies a shared centered network with domain-specific weights in a star topology, while M2M (Zhang et al., 2022) and APG (Yan et al., 2022) generate customized network parameters from scenario or instance context. Recent mixture-of-experts variants such as STEM (Su et al., 2024) and HoME (Wang et al., 2025) push specialization further via task-specific embeddings, stop-gradient gating, and hierarchical expert grouping. Orthogonally, an optimization-centric line attributes multi-task interference to conflicting gradients on shared parameters and manipulates the update direction, e.g., PCGrad (Yu et al., 2020), GradNorm (Chen et al., 2018), CAGrad (Liu et al., 2021) and Nash-MTL (Navon et al., 2022).

5.2 MTL in GR

Generative recommendation (GR) reformulates recommendation as sequential transduction over user behaviors, with backbones such as HSTU (Zhai et al., 2024), TIGER (Rajput et al., 2023), and OneRec (Deng et al., 2025) scaling attention-based next-item generation. Multi-task modeling in GR, however, remains largely inherited from the classical recipe. Tasks are still read out of one shared sequence representation. IntTravel (Yan et al., 2026) strengthens an HSTU backbone for travel recommendation but encodes the behavior sequence without task tokens, branching only after encoding. OneTrans (Zhang et al., 2026) unifies sequential and non-sequential features into a single token stream with mixed shared/token-specific parameterization, yet it likewise carries no task token and follows the unified single-stream paradigm. HGenPush (Liang et al., 2026) and Onerank (Tang et al., 2026) takes a first step toward task-aware encoding by inserting task tokens into the sequence, but these tokens share one stream and one parameter set with the context tokens, task-specific signals are still diluted in the shared space.

6 Conclusion

We revisit multi-task generative recommendation through the lens of a threefold collapse inherent in the prevailing recipe of a single task-agnostic representation with conditionally independent heads. We propose **IntHQ**, which counters each collapse with a dedicated component. Dual-Stream Decoupling injects task signals early and separates the task stream from the context stream in parameter space. Task-Interactive Modeling learns input-dependent tasks interactions rather than a prescribed structure. Hierarchical Querying lets each task retrieve information across representations at different scale. In online A/B tests on Amap serving hundreds of millions of users, **IntHQ** delivers a 1.60% UVCTR relative lift. Experiments across a grid of encoders and multi-task heads confirm that the gains stem from moving task signals into the encoding stage and persist across heterogeneous head architectures. This work offers a practical way to scale generative recommender systems in complex multi-task setting.

References

- Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013.
- Rich Caruana. Multitask learning. *Machine learning*, 28(1):41–75, 1997.
- Zhao Chen, Vijay Badrinarayanan, Chen-Yu Lee, and Andrew Rabinovich. Gradnorm: Gradient normalization for adaptive loss balancing in deep multitask networks. In *International conference on machine learning*, pp. 794–803. PMLR, 2018.
- Zhiyuan Cheng, James Caverlee, Kyumin Lee, and Daniel Sui. Exploring millions of footprints in location sharing services. In *Proceedings of the International AAAI Conference on Web and Social Media*, pp. 81–88, 2011.
- Eunjoon Cho, Seth A Myers, and Jure Leskovec. Friendship and mobility: user movement in location-based social networks. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 1082–1090, 2011.
- Thomas M Cover. *Elements of information theory*. John Wiley & Sons, 1999.
- Jiaxin Deng, Shiyao Wang, Kuo Cai, Lejian Ren, Qigen Hu, Weifeng Ding, Qiang Luo, and Guorui Zhou. Onerec: Unifying retrieve and rank with generative recommender and iterative preference alignment. *arXiv preprint arXiv:2502.18965*, 2025.
- Tilmann Gneiting and Adrian E Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American statistical Association*, 102(477):359–378, 2007.
- Min Hou, Le Wu, Yuxin Liao, Yonghui Yang, Zhen Zhang, Yu Wang, Changlong Zheng, Han Wu, and Richang Hong. A survey on generative recommendation: Data, model, and tasks. *AI Open*, 2026.
- Xiao Liang, Jiali Feng, Xin Feng, Yiqing Wang, Baolin Ye, Siyao Feng, Zhihui Deng, Cunyi Zhang, Huajin Sun, Xuanping Li, et al. Hgenpush: A heterogeneous generative recommendation architecture for industrial push notification systems. *arXiv preprint arXiv:2607.03362*, 2026.
- Bo Liu, Xingchao Liu, Xiaojie Jin, Peter Stone, and Qiang Liu. Conflict-averse gradient descent for multi-task learning. *Advances in neural information processing systems*, 34: 18878–18890, 2021.
- Jiaqi Ma, Zhe Zhao, Xinyang Yi, Jilin Chen, Lichan Hong, and Ed H Chi. Modeling task relationships in multi-task learning with multi-gate mixture-of-experts. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 1930–1939, 2018a.
- Xiao Ma, Liqin Zhao, Guan Huang, Zhi Wang, Zelin Hu, Xiaoqiang Zhu, and Kun Gai. Entire space multi-task model: An effective approach for estimating post-click conversion rate. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pp. 1137–1140, 2018b.
- Ishan Misra, Abhinav Shrivastava, Abhinav Gupta, and Martial Hebert. Cross-stitch networks for multi-task learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3994–4003, 2016.
- Giovanni Monea, Nathan Godey, Kianté Brantley, and Yoav Artzi. The state-prediction separation hypothesis. *arXiv preprint arXiv:2607.01218*, 2026.
- Diego Monti, Enrico Palumbo, Giuseppe Rizzo, Raphaël Troncy, Thibault Ehrhart, and Maurizio Morisio. Semantic trails of city explorations: How do we live a city. *arXiv preprint arXiv:1812.04367*, 2018.

- Aviv Navon, Aviv Shamsian, Idan Achituve, Haggai Maron, Kenji Kawaguchi, Gal Chechik, and Ethan Fetaya. Multi-task learning as a bargaining game. *arXiv preprint arXiv:2202.01017*, 2022.
- Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.
- Shashank Rajput, Nikhil Mehta, Anima Singh, Raghunandan Hulikal Keshavan, Trung Vu, Lukasz Heldt, Lichan Hong, Yi Tay, Vinh Q Tran, Jonah Samost, et al. Recommender systems with generative retrieval. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.
- Xiang-Rong Sheng, Liqin Zhao, Guorui Zhou, Xinyao Ding, Binding Dai, Qiang Luo, Siran Yang, Jingshan Lv, Chi Zhang, Hongbo Deng, et al. One model to serve all: Star topology adaptive recommender for multi-domain ctr prediction. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pp. 4104–4113, 2021.
- Teng Shi, Zihua Si, Jun Xu, Xiao Zhang, Xiaoxue Zang, Kai Zheng, Dewei Leng, Yanan Niu, and Yang Song. Unisar: Modeling user transition behaviors between search and recommendation. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 1029–1039, 2024.
- Teng Shi, Weicong Qin, Weijie Yu, Xiao Zhang, Ming He, Jianping Fan, and Jun Xu. Mining informative interests via latent cross reasoning for search enhanced recommendation. In *Proceedings of the 49th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 1612–1622, 2026.
- Liangcai Su, Junwei Pan, Ximei Wang, Xi Xiao, Shijie Quan, Xihua Chen, and Jie Jiang. Stem: unleashing the power of embeddings for multi-task recommendation. In *Proceedings of the AAAI conference on artificial intelligence*, pp. 9002–9010, 2024.
- Ximeng Sun, Rameswar Panda, Rogerio Feris, and Kate Saenko. Adashare: Learning what to share for efficient deep multi-task learning. *Advances in Neural Information Processing Systems*, 33:8728–8740, 2020.
- Hongyan Tang, Junning Liu, Ming Zhao, and Xudong Gong. Progressive layered extraction (ple): A novel multi-task learning (mtl) model for personalized recommendations. In *Proceedings of the 14th ACM conference on recommender systems*, pp. 269–278, 2020.
- Jiakai Tang, Sunhao Dai, Kun Wang, Zhiluohan Guo, Yu Zhao, Cong Fu, Kangle Wu, Yabo Ni, Anxiang Zeng, Xu Chen, et al. Onerank: Unified transformer-native ranking architecture for multi-task recommendation. *arXiv preprint arXiv:2606.16838*, 2026.
- Pantelis Vafidis, Aman Bhargava, and Antonio Rangel. Disentangling representations through multi-task learning. In *International Conference on Learning Representations*, volume 2025, pp. 68296–68338, 2025.
- Greg Ver Steeg and Aram Galstyan. Discovering structure in high-dimensional data through correlation explanation. *Advances in neural information processing systems*, 27, 2014.
- Xu Wang, Jiangxia Cao, Zhiyi Fu, Kun Gai, and Guorui Zhou. Home: Hierarchy of multi-gate experts for multi-task learning at kuaishou. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 1*, pp. 2638–2647, 2025.
- Yuhao Wang, Ha Tsz Lam, Yi Wong, Ziru Liu, Xiangyu Zhao, Yichao Wang, Bo Chen, Huifeng Guo, and Ruiming Tang. Multi-task deep recommender systems: A survey. *arXiv preprint arXiv:2302.03525*, 2023.
- Satosi Watanabe. Information theoretical analysis of multivariate correlation. *IBM Journal of research and development*, 4(1):66–82, 1960.

- Dongbo Xi, Zhen Chen, Peng Yan, Yinger Zhang, Yongchun Zhu, Fuzhen Zhuang, and Yu Chen. Modeling the sequential dependence among audience multi-step conversions with multi-task learning in targeted display advertising. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pp. 3745–3755, 2021.
- Bencheng Yan, Pengjie Wang, Kai Zhang, Feng Li, Hongbo Deng, Jian Xu, and Bo Zheng. Appg: Adaptive parameter generation network for click-through rate prediction. *Advances in Neural Information Processing Systems*, 35:24740–24752, 2022.
- Huimin Yan, Longfei Xu, Junjie Sun, Ni Ou, Wei Luo, Xing Tan, Ran Cheng, Kaikui Liu, and Xiangxiang Chu. Intsr: An integrated generative framework for search and recommendation. *arXiv preprint arXiv:2509.21179*, 2025.
- Huimin Yan, Longfei Xu, Junjie Sun, Zheng Liu, Wei Luo, Kaikui Liu, and Xiangxiang Chu. Intravel: A real-world dataset and generative framework for integrated multi-task travel recommendation. *arXiv preprint arXiv:2602.11664*, 2026.
- Dingqi Yang, Daqing Zhang, Vincent W Zheng, and Zhiyong Yu. Modeling user activity preference by leveraging user spatial temporal characteristics in lbsns. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 45(1):129–142, 2014.
- Dingqi Yang, Daqing Zhang, and Bingqing Qu. Participatory cultural mapping based on collective behavior data in location-based social networks. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 7(3):1–23, 2016.
- Dingqi Yang, Bingqing Qu, Jie Yang, and Philippe Cudre-Mauroux. Revisiting user mobility and social relationships in lbsns: a hypergraph embedding approach. In *The world wide web conference*, pp. 2147–2157, 2019.
- Jiahao Yu, Yihai Duan, Longfei Xu, Chao Chen, Shuliang Liu, Kaikui Liu, Fan Yang, Xiangxiang Chu, and Ning Guo. Dsfnet: Learning disentangled scenario factorization for multi-scenario route ranking. *arXiv preprint arXiv:2404.00243*, 2024.
- Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. Gradient surgery for multi-task learning. *Advances in neural information processing systems*, 33:5824–5836, 2020.
- Jiaqi Zhai, Lucy Liao, Xing Liu, Yueming Wang, Rui Li, Xuan Cao, Leon Gao, Zhaojie Gong, Fangda Gu, Michael He, et al. Actions speak louder than words: Trillion-parameter sequential transducers for generative recommendations. *arXiv preprint arXiv:2402.17152*, 2024.
- Mingzhu Zhang, Ruiping Yin, Zhen Yang, and Yipeng Wang. Advances and challenges of multi-task learning method in recommender systems: A survey. *Neurocomputing*, pp. 132510, 2025.
- Qianqian Zhang, Xinru Liao, Quan Liu, Jian Xu, and Bo Zheng. Leaving no one behind: A multi-scenario multi-task meta learning approach for advertiser modeling. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*, pp. 1368–1376, 2022.
- Zhaoqi Zhang, Haolei Pei, Jun Guo, Tianyu Wang, Yufei Feng, Hui Sun, Shaowei Liu, and Aixin Sun. Onetrans: Unified feature interaction and sequence modeling with one transformer in industrial recommender. In *Proceedings of the ACM Web Conference 2026*, pp. 8162–8170, 2026.

A Proofs and Extended Statements for the Pilot Study

A.1 Source collapse motivates Dual-Stream Decoupling.

Two statements settle the task signal must enter and on which parameters it must live. The first rules out the standard remedy.

Proposition 1 (Late conditioning is powerless). *If every task-specific computation is a function of the shared code, $h_k = g_k(z)$, then $\sum_k I(\mathcal{A}; y_k | h_k) \geq \Delta_{\text{src}}(z)$, with equality if and only if each g_k is sufficient for y_k relative to z .*

Nothing placed above z can reduce the source term, because each module only coarsens a statistic from the discarded information. Task conditioning must therefore happen inside the encoder. Proposition 2 shows that placing it inside a stream whose weights are shared with the context computation is still not enough.

Proposition 2 (Price of a shared block). *Let a weight block act in two roles, and write the risk on two copies of its parameters as $R(\Theta_c, \Theta_q) = L_c(\Theta_c) + L_q(\Theta_q)$, the additive form of the gradient split $\nabla_{\Theta} R = g_c + g_q$. Let L_c, L_q have minimizers Θ_c^*, Θ_q^* with positive-definite Hessians H_c, H_q , and let Θ^t minimize the tied risk $L_c(\Theta) + L_q(\Theta)$. Then at Θ^t the two roles pull with equal and opposite force, $\nabla L_c(\Theta^t) = -\nabla L_q(\Theta^t)$, so neither role sits at its own optimum, and the tied risk exceeds the untied risk $L_c(\Theta_c^*) + L_q(\Theta_q^*)$ by*

$$\frac{1}{2} \Delta^\top H_c (H_c + H_q)^{-1} H_q \Delta \geq 0, \quad \Delta := \Theta_c^* - \Theta_q^*, \quad (11)$$

which vanishes if and only if the two role optima coincide.

The cost (11) depends only on how far apart the two roles want the block to be, not on how many parameters it has. When they disagree, sharing forces a curvature-weighted compromise that serves neither. Dual-Stream Decoupling answers both propositions. It conditions the computation on the task from the first layer, so the task identity is present before the compression is complete, and it keeps the task stream on parameter-decoupled weights, so each role reaches its own optimum. This turns the state and prediction separation of (Monea et al., 2026), established for single-task language modeling, into multi-task case with K task roles against one context role.

A.2 Relation collapse motivates Task-Interactive Modeling.

Proposition 3 (Factorization gap and the escape routes). *On a fixed encoder, replacing the best factorized read-out by the best joint read-out reduces the risk by exactly $\text{TC}(\mathbf{y} | z)$, and*

$$\text{TC}(\mathbf{y} | z) - \text{TC}(\mathbf{y} | \mathcal{A}) = \sum_k I(\mathcal{A}; y_k | z) - I(\mathcal{A}; \mathbf{y} | z), \quad (12)$$

so the realized cost and the encoder-free floor coincide only at joint sufficiency. Attaining R^* requires a read-out that is not factorized given $\mathcal{A}$, and there are exactly three ways to obtain one. An explicit joint head over $\prod_k \mathcal{I}_k$, a latent-variable head scored by its marginal likelihood, or a chain-rule factorization in which each decision conditions on realized predecessor outcomes.

Proposition 4 (Residual under realized-outcome conditioning). *Let $\prec$ be a linear extension of the funnel order and let each head condition on the realized labels of $\text{pa}(k) \subseteq \{y_j : j \prec k\}$. Then the optimal excess risk over R^* equals*

$$\sum_k I(y_k; y_{\prec k} \setminus \text{pa}(k) \mid y_{\text{pa}(k)}, \mathcal{A}), \quad (13)$$

which is invariant to the choice of extension, vanishes when every parent set is the full predecessor set, and equals Δ_{rel} when all parent sets are empty.

Task-Interactive Modeling places each decision at the position where its outcome is realized inside a session, and masks the reverse direction. Funnel models such as ESM (Ma et al., 2018b) and AITM (Xi et al., 2021) also condition on predecessors, yet they fix the order and the coupling strength in advance, whereas cross-task attention learns that strength for each instance.

A.3 Gibbs optimum and the value of R_{TAE}

Lemma 1 (Gibbs inequality (Cover, 1999)). *Let y be a target with finitely possible values and let u collect everything the predictor is allowed to observe. Write $p(\cdot | u)$ for the true*

conditional distribution of y given u , and let $q(\cdot | u)$ be any predicted distribution that gives positive probability to every value, as a softmax head does. Then the expected log loss of q splits into two parts,

$$\underbrace{\mathbb{E}[-\log q(y | u)]}_{\text{loss of the prediction}} = \underbrace{H(y | u)}_{\text{uncertainty of the truth}} + \underbrace{\mathbb{E}_u D_{\text{KL}}(p(\cdot | u) \| q(\cdot | u))}_{\text{distance from the truth}} \geq H(y | u). \quad (14)$$

The second part is zero exactly when $q(\cdot | u) = p(\cdot | u)$. The log loss is therefore minimized by predicting the true conditional distribution and by nothing else, which is the statement that it is a strictly proper scoring rule ([Gneiting & Raftery, 2007](#)).

Proof. We give the short derivation for completeness. For one outcome, add and subtract the log probability of the truth,

$$-\log q(y | u) = -\log p(y | u) + \log \frac{p(y | u)}{q(y | u)}.$$

Now average both sides over y and u . The first term averages to $H(y | u)$, which is the definition of conditional entropy, and the second averages to $\mathbb{E}_u D_{\text{KL}}(p \| q)$, which is the definition of the divergence. This gives the identity in (14). $\square$

Lemma 2 (Optimal risk of a factorized read-out). *Suppose each head is free to output any distribution over its own candidate set and no parameter is shared between heads, the K heads can be chosen independently of one another. Then*

$$R_{\text{TAE}}(z) := \inf_{q_1, \dots, q_K} \mathbb{E} \left[-\sum_k \log q_k(y_k | z) \right] = \sum_k H(y_k | z), \quad (15)$$

reached by letting every head output the true conditional distribution $q_k = p(y_k | z)$. Since $-\sum_k \log q_k(y_k | z) = -\log \prod_k q_k(y_k | z)$, the left side is the joint log loss of the product law $\prod_k q_k(\cdot | z)$, which is what makes $R_{\text{TAE}}(z)$ comparable with R^* .

A.4 Proof of Theorem 1

Proof. Both R^* and $R_{\text{TAE}}(z)$ are values of the same joint log loss: Lemma 1 with target $\mathbf{y}$ and $u = \mathcal{A}$ gives $R^* = H(\mathbf{y} | \mathcal{A})$, and Lemma 2 gives $R_{\text{TAE}}(z) = \sum_k H(y_k | z)$.

Because z is a deterministic function of $\mathcal{A}$, conditioning on $(z, \mathcal{A})$ is the same as conditioning on $\mathcal{A}$, so $H(y_k | z) = H(y_k | \mathcal{A}) + I(\mathcal{A}; y_k | z)$; all terms are finite. Summing over k and using the definition of total correlation, $\sum_k H(y_k | \mathcal{A}) = H(\mathbf{y} | \mathcal{A}) + \text{TC}(\mathbf{y} | \mathcal{A})$,

$$R_{\text{TAE}}(z) = \underbrace{\sum_k I(\mathcal{A}; y_k | z)}_{\Delta_{\text{src}}(z)} + \underbrace{H(\mathbf{y} | \mathcal{A}) + \text{TC}(\mathbf{y} | \mathcal{A})}_{\Delta_{\text{rel}}},$$

which is (4) after subtracting $R^* = H(\mathbf{y} | \mathcal{A})$. Both terms are non-negative, $\Delta_{\text{src}}(z)$ as a sum of conditional mutual informations and $\Delta_{\text{rel}} = \mathbb{E}_{\mathcal{A}} D_{\text{KL}}(P(\mathbf{y} | \mathcal{A}) \| \prod_k P(y_k | \mathcal{A}))$ as an expected divergence. Dropping $\Delta_{\text{src}}(z)$ gives the floor $R_{\text{TAE}}(z) \geq R^* + \Delta_{\text{rel}}$, with equality exactly when $I(\mathcal{A}; y_k | z) = 0$ for all k , that is when z is marginally sufficient for every task. $\square$

A.5 Source term

A.5.1 Late conditioning cannot reduce the source term

Proof of Proposition 1. Let $h_k = g_k(z)$ with g_k measurable. We show the inequality term by term.

First, $\sigma(h_k) \subseteq \sigma(z)$, because h_k is a measurable function of z . Conditioning on a coarser σ -field cannot decrease conditional entropy, hence

$$H(y_k | h_k) \geq H(y_k | z). \quad (16)$$

Second, h_k is also a measurable function of $\mathcal{A}$, since z is, so Step 2 of the proof of Theorem 1 applies with h_k in place of z and gives $I(\mathcal{A}; y_k | h_k) = H(y_k | h_k) - H(y_k | \mathcal{A})$. Combining this with (16) and with Step 3 of that proof,

$$I(\mathcal{A}; y_k | h_k) = H(y_k | h_k) - H(y_k | \mathcal{A}) \geq H(y_k | z) - H(y_k | \mathcal{A}) = I(\mathcal{A}; y_k | z).$$

Summing over k gives the claim. Equality holds for a given k if and only if (16) is tight, that is if and only if h_k retains all the information about y_k that z carries. $\square$

A.5.2 The price of a shared block

A weight block W that a Transformer applies at every position acts in two roles. At context positions it prepares keys and values read by later positions, the context role; at task positions it produces the read-out for a decision, the task role. By the chain rule its gradient splits by source,

$$\nabla_W R = \underbrace{\sum_{p \in \mathcal{C}} \nabla_{W_p} R}_{g_c} + \underbrace{\sum_{p \in \mathcal{P}} \nabla_{W_p} R}_{g_q}, \quad (17)$$

where $\mathcal{C}$ and $\mathcal{P}$ index context and task positions and W_p is the copy of W used at p . This is the multi-task form of the state and prediction split of (Monea et al., 2026), the context role playing the state role and the K decisions playing K prediction roles. Integrating (17) along the two paths, we model the risk on two formal copies of the block as additive,

$$R(\Theta_c, \Theta_q) = L_c(\Theta_c) + L_q(\Theta_q), \quad (18)$$

with Θ_c used wherever the block acts in the context role and Θ_q wherever it acts in the task role. The tied architecture imposes $\Theta_c = \Theta_q$; the decoupled one does not.

Proof of Proposition 2. The untied problem separates by (18) and attains $L_c(\Theta_c^*) + L_q(\Theta_q^*)$. The tied problem minimizes $J(\Theta) = L_c(\Theta) + L_q(\Theta)$; its stationarity condition $\nabla L_c(\Theta^t) + \nabla L_q(\Theta^t) = 0$ gives

$$\nabla L_c(\Theta^t) = -\nabla L_q(\Theta^t), \quad (19)$$

so, unless both gradients vanish, which requires a common minimizer, the two roles pull with equal and opposite force and neither sits at its own optimum. This is the first claim.

For the magnitude, model each role loss near its minimizer by $L_r(\Theta) = L_r(\Theta_r^*) + \frac{1}{2}(\Theta - \Theta_r^*)^\top H_r(\Theta - \Theta_r^*)$ for $r \in \{c, q\}$, with $H_c, H_q \succ 0$. Minimizing the sum of two quadratics is a standard identity: for $A, B \succ 0$,

$$\min_{\Theta} \frac{1}{2}(\Theta - a)^\top A(\Theta - a) + \frac{1}{2}(\Theta - b)^\top B(\Theta - b) = \frac{1}{2}(a - b)^\top (A^{-1} + B^{-1})^{-1}(a - b), \quad (20)$$

attained at $\Theta = (A + B)^{-1}(Aa + Bb)$. Applying (20) with $a = \Theta_c^*, b = \Theta_q^*, A = H_c, B = H_q$ gives the location $\Theta^t = (H_c + H_q)^{-1}(H_c \Theta_c^* + H_q \Theta_q^*)$ and the excess

$$J(\Theta^t) - [L_c(\Theta_c^*) + L_q(\Theta_q^*)] = \frac{1}{2} \Delta^\top (H_c^{-1} + H_q^{-1})^{-1} \Delta, \quad \Delta = \Theta_c^* - \Theta_q^*.$$

The combined matrix $(H_c^{-1} + H_q^{-1})^{-1} = H_c(H_c + H_q)^{-1}H_q$ is symmetric positive definite, so the excess is non-negative and vanishes exactly when $\Delta = 0$, which is (11). $\square$

A.6 Relation term

Proof of Proposition 3. The gap on a fixed encoder. By Lemma 1 with target $\mathbf{y}$ and $u = z$, the best joint read-out on z attains $H(\mathbf{y} | z)$. By Lemma 2 the best factorized read-out attains $\sum_k H(y_k | z)$. The difference is $\sum_k H(y_k | z) - H(\mathbf{y} | z) = \text{TC}(\mathbf{y} | z)$, which by the same divergence identity used in Step 5 of Theorem 1, now conditioned on z , equals $\mathbb{E}_z D_{\text{KL}}(P(\mathbf{y} | z) \| \prod_k P(y_k | z)) \geq 0$ and vanishes exactly under conditional independence given z .

The relation between the two total correlations. Subtract the two definitions and apply Step 3 of Theorem 1 once to each marginal and once to the joint,

$$\text{TC}(\mathbf{y} | z) - \text{TC}(\mathbf{y} | \mathcal{A}) = \sum_k [H(y_k | z) - H(y_k | \mathcal{A})] - [H(\mathbf{y} | z) - H(\mathbf{y} | \mathcal{A})]$$

which is (12). Neither side dominates in general, since the first sum counts information that several decisions share once per decision while the second counts it once, and both sides vanish together when z is jointly sufficient. $\square$

For any total order $\prec$ that extends the funnel order, the entropy chain rule gives $H(\mathbf{y} | \mathcal{A}) = \sum_k H(y_k | y_{\prec k}, \mathcal{A})$, hence

$$\text{TC}(\mathbf{y} | \mathcal{A}) = \sum_k I(y_k; y_{\prec k} | \mathcal{A}), \quad (21)$$

which we read as the total value of conditioning each decision on its predecessors.

Proof of Proposition 4. Assume that the parent assignment pa induces an acyclic graph and that $\prec$ is one of its linear extensions, so $\text{pa}(k) \subseteq \{y_j : j \prec k\}$ for every k and $\prod_k q_k(y_k | y_{\text{pa}(k)}, \mathcal{A})$ is a normalized joint law. Acyclicity is required, since with a cyclic assignment the product is only a pseudo-likelihood, Lemma 1 does not apply to it, and the resulting excess can be negative.

Step 1, the optimum of the parent-conditioned family. Apply Lemma 1 to the k -th factor with target y_k and conditioning variable $u = (y_{\text{pa}(k)}, \mathcal{A})$. As in Lemma 2 the objective is a sum whose k -th summand depends on q_k only, so the infimum splits and equals $\sum_k H(y_k | y_{\text{pa}(k)}, \mathcal{A})$.

Step 2, subtracting the baseline term by term. Write $R^* = \sum_k H(y_k | y_{\prec k}, \mathcal{A})$, which is the chain rule along $\prec$. Subtracting term by term,

$$\sum_k [H(y_k | y_{\text{pa}(k)}, \mathcal{A}) - H(y_k | y_{\prec k}, \mathcal{A})] = \sum_k I(y_k; y_{\prec k} \setminus \text{pa}(k) | y_{\text{pa}(k)}, \mathcal{A}),$$

where each equality uses $\text{pa}(k) \subseteq \{y_j : j \prec k\}$, so that passing from the conditioning set $(y_{\text{pa}(k)}, \mathcal{A})$ to the larger set $(y_{\prec k}, \mathcal{A})$ removes exactly the variables $y_{\prec k} \setminus \text{pa}(k)$ and the entropy difference is by definition the conditional mutual information with those variables. Every term is non-negative, so the total is non-negative.

Step 3, invariance to the extension. The total equals $\sum_k H(y_k | y_{\text{pa}(k)}, \mathcal{A}) - H(\mathbf{y} | \mathcal{A})$. The first sum depends only on the fixed parent assignment and not on $\prec$, and the second is order free, so the total is the same for every linear extension of the same graph, even though the individual terms in Step 2 do depend on the extension.

Step 4, the two boundary cases. Taking $\text{pa}(k) = \{y_j : j \prec k\}$ makes $y_{\prec k} \setminus \text{pa}(k) = \emptyset$, so every term is zero and the total vanishes. Taking every $\text{pa}(k) = \emptyset$ turns the total into $\sum_k I(y_k; y_{\prec k} | \mathcal{A})$, which is $\text{TC}(\mathbf{y} | \mathcal{A})$ by (21). $\square$

B Computation Complexity

Let N_s be the number of sessions, N_c the number of context tokens, $N_q = KN_s$ the number of task tokens (K tasks per session), d the width, L the depth.

A unified single-stream encoder that places task tokens in the same sequence attends over $N_c + N_q$ positions, costing $\mathcal{O}(L(N_c + N_q)^2 d)$ for attention. **IntHQ** instead performs three attentions per layer, i.e., context self-attention, task $\rightarrow$ context cross-attention, and task self-attention. For a total of:

$$\mathcal{O}(L(N_c^2 + N_q N_c + N_q^2) d), \quad (22)$$

which is *strictly cheaper* by $(N_c + N_q)^2$. Finally, read-out differs asymptotically: a factorized head on a unified stream is evaluated at all $N_c + N_q$ positions, whereas in **IntHQ** each

Table 3: Measured cost with the task head fixed to STAR ($L=3$, $d=96$, batch size 1). Best in **bold**.

Encoder	Params	FLOPs
IntTravel	1.54M	448.8M
OneTrans	3.43M	432.3M
HGenPush	1.71M	561.6M
IntHQ (Ours)	1.70M	297.9M

task head consumes only its own N_s task tokens, reducing head cost from $\mathcal{O}(KNd_{\text{head}})$ to $\mathcal{O}(KN_sd_{\text{head}})$. Table 3 reports measured cost with the head fixed to STAR. We count dense parameters of the encoder, read-out gate and head (excluding sparse embedding tables) and operator-level FLOPs.