PerDucer: Keyphrase-Driven Personalization Inducer for Summarization from User Histories

Parthiv Chatterjee, Asish Joel Batha, Sourish Dasgupta
KDM Lab, Dhirubhai Ambani University, Gandhinagar, Gujarat, India
202421011@dau.ac.in asishjoel.b@gmail.com sourish_dasgupta@dau.ac.in

Tanmoy Chakraborty
IIT Delhi, India; IIT Delhi Abu Dhabi, UAE
tanchak@iitd.ac.in

Abstract

Document summarization becomes more challenging when summaries must reflect a user's subjective interests in addition to document salience. SOTA Large Language Models (LLMs) show strong in-context summarization capabilities. Prior works report that simply prepending long and dynamically evolving user histories leads to unstable, inconsistent personalization. To address this, we introduce PerDucer, a personalization inducer for frozen language models. Given a user interaction sequence (trajectory) and a query document, PerDucer first predicts the next likely preference signal. It then maps the latent signal to a small set of personalized keyphrases for the query document. These keyphrases serve as the control cues that steer the frozen summarizers (both LLMs and SLMs) towards user-aligned summaries. Across PENS and a reconstructed temporal version of OpenAI-Reddit, PerDucer-boosted LLMs outperform the strongest history-prompting baselines, yielding an average +0.18 improvement across personalization metrics (PerSEval). Two PerDucer-augmented SLMs approach the top-performing evaluated LLM, with SmolLM2-1.7B reaching ≈97% of the best-performing DeepSeek-R1-14B score. These results suggest that short keyphrase cues can induce personalization in frozen summarizers without modifying their parameters. The codes are available at https://github.com/KDM-LAB/PerDucer-ACL_2026_Findings.git.

1 Introduction

Modern summarizers reduce the cost of consuming frequent updates. Yet personalized summarization is challenging because summaries must reflect user-specific interests in addition to document salience, especially for multi-aspect documents (Dasgupta et al., 2024). Most prior work models personalization through static attributes or coarse topical interests (Dou et al., 2021a; He et al., 2022a; Li et al., 2023). In temporal settings, preferences evolve at subtopic granularity, as reflected in MS/CAS PENS (Ao et al., 2021). Appending long interaction histories to LLM prompts does not yield stable personalization, and performance can decline even under fixed prompt length (Chen et al., 2025; Gao et al., 2024; Patel et al., 2024).

We reformulate history-injected personalization as keyphrase-guided personalization, and introduce PerDucer, a keyphrase-driven *Personalization Inducer*. Rather than forcing LLMs to process entire user histories, PerDucer distills a user's evolving reading behavior into a compact, interpretable set of *personalized keyphrases* tailored to the query document. These keyphrases act as lightweight control cues for frozen summarizers without parameter updates. PerDucer operates over a temporal user-interaction representation that captures documents, summaries, and user actions such as click, skip, or summary-read. Its encoder predicts the user's next latent preference signal conditioned on the query document. A decoder then maps this signal to a ranked keyphrase list, which is inserted into simplified ICL prompts for LLMs or prepended to the input for non-LLM summarizers.

We evaluate PerDucer through three research questions: **RQ-1:** Can PerDucer boost personalization in strong *LLMs*? **RQ-2:** Can it elevate *small language models* (SLMs) to near-LLM personalization performance? and **RQ-3:** Can it enable *vanilla summarizers* to outperform specialized personalized summarizers that condition on user history?

We evaluate PerDucer on PENS (Ao et al.,

Findings of the Association for Computational Linguistics: ACL 2026, pages 20651–20677

2021) and a synthetic trajectory reconstruction of OpenAI-Reddit, using PerSEval variants PSE-JSD, PSE-SU4, and PSE-METEOR (Dasgupta et al., 2024). Across four frozen LLMs, PerDucer improves PSE metrics by 0.1, 0.05, and 0.23 over history-prompting baselines (Jiang et al., 2023; Tunstall et al., 2023; DeepSeek-AI et al., 2025; Touvron et al., 2023). Two SLMs, SmolLM2-1.7B (Allal et al., 2025) and Qwen2.5-0.5B (Qwen et al., 2025), reach near-LLM behavior. We observe that SmolLM2 achieves **97.02%** of DeepSeek-14B's personalization score. Finally, when injected into vanilla summarizers such as BigBird-Pegasus (Zaheer et al., 2020) and SimCLS (Liu and Liu, 2021), PerDucer surpasses the strongest specialized personalized system, GTP (Song et al., 2023), with gains of **0.18/0.09/0.11**↑. These findings suggest that reframing personalization as **keyphrase-guided induction** is both effective and model-agnostic, enabling any frozen summarizers (large or small) to adapt finely to evolving user interests.

2 Background

Personalized Summarization. Personalized summarization aims to generate document summaries that align with a reader's evolving preferences rather than global salience. Standard summarization metrics like Rouge (Lin, 2004), METEOR (Banerjee and Lavie, 2005), or BertScore (Zhang et al., 2020) measure specifically the content selection fidelity, but fails to evaluate how well a system captures user-specific expectations inferred from temporal behaviors like *click*, *skip*, or *summarize*.

Training and Evaluation Datasets. Temporal personalization datasets require (i) an ordered interaction history, (ii) user-specific references for shared documents, and (iii) sufficient topical diversity to capture preference shifts. Large summarization datasets such as CNN/DM (Hermann et al., 2015) or MultiNews (Fabbri et al., 2019) lack user-specific references. In contrast, OpenAI-Reddit (Völske et al., 2017) lacks time-stamped interaction histories. Only PENS (Ao et al., 2021) and PersonalSum (Zhang et al., 2024) satisfy all criteria. PENS training data contains impression logs rather than user-specific summary nodes in training. We therefore augment PENS trajectories with surrogate summary nodes so that click/skip histories can be converted into trajectory-level supervision, following (Chatterjee et al., 2025). OpenAI-Reddit lacks native temporal histories. Thus, we reconstruct user trajectories from confidence-filtered ratings and selected summaries, again following (Chatterjee et al., 2025). We further apply trajectory-level augmentation during training to diversify these reconstructed sequences. This augmentation increases the topical diversity on PENS, with an average 13.6 topics per trajectory and a topic-change rate of 0.77. We also derive synthetic temporal orders for OpenAI-Reddit following (Chatterjee et al., 2025).

Personalized Guided Summarization. Most personalized summarization approaches rely on either static user persona or document-only control signals. *GSUM* injects user-provided keyphrases extracted from the document and does not condition the extracted control signals on interaction histories (Dou et al., 2021b), *CTRLSum*, *TMWIN*, and *Tri-Agent* apply static control codes, topic markers, or fixed editing preferences rather than trajectory-conditioned control codes (He et al., 2022b; Kirstein et al., 2024; Xiao et al., 2024). PENS introduced user-encoder-based personalized summarizer which injects a user embedding derived from clicked-history encoders into a decoder (Ao et al., 2021). The GTP model (Song et al., 2023) extracts latent editing operators from user trajectories, while Signature-Phrase (Cai et al., 2023) compresses trajectories into keyphrases.

Thus, existing approaches assume static personas, ignore action-level heterogeneity, or compress behavior without capturing temporal evolution. To our knowledge, prior personalized summarization systems do not explicitly induce query-conditioned keyphrase controls from dynamic interaction trajectories. This gap motivates keyphrase-driven personalization inducer, PerDucer.

3 Personalized Summarization: Formulation

A crucial distinction in personalized summarization is between a *static user persona* and a *dynamic user-preference history*. Static persona attributes. such as nationality or coarse topical interests, are approximately time-invariant. By contrast, preference histories reflect user behavior as a temporal sequence over multiple documents and interactions that evolve at subtopic granularity. Methods that condition only on static personas cannot represent the fine-grained temporal shifts observed in realistic datasets such as PENS (Ao et al., 2021). Following Chatterjee et al. (2025), we adopt the User–Interaction Graph (UIG) formalism as a rep-

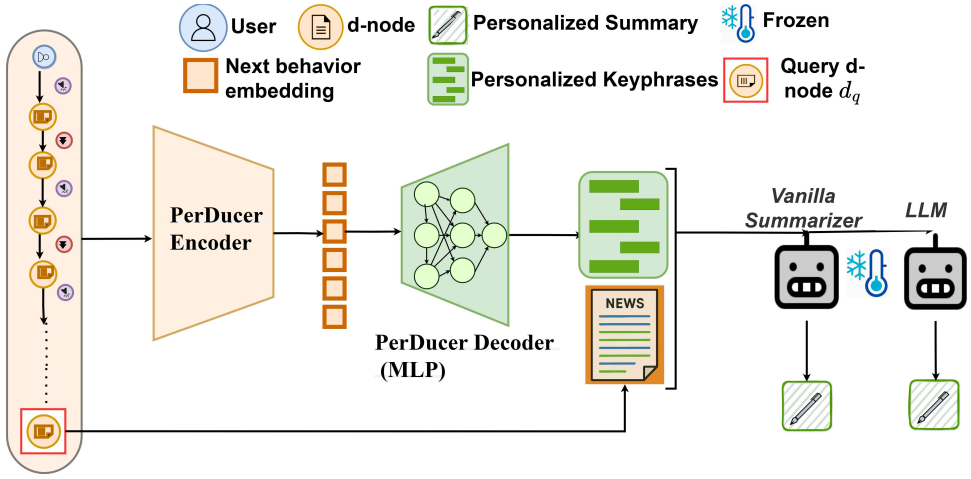

Figure 1: **PerDucer Pipeline.** The PerDucer encoder encodes the user trajectory and predicts the next-step user behavior embedding, which is decoded by an MLP-based key-phrase extractor. The top-k predicted key phrases are injected as control cues into frozen summarization models.

resentation of dynamic user histories. In this representation, UIG is a DAG G = ⟨N, E⟩ with three disjoint node types: **u-nodes** $u^{(t_0)}$ representing users at the initial time point t_0, **d-nodes** $d^{(t_p)}$ representing documents the user interacted with at time t_p, and **s-nodes** $s_j^{(t_q)}$ representing user-specific summaries (either user-written or model-generated) at time t_q, associated with a previously viewed document at t_{q-1}. Edges in E encode user actions, which are document-level interactions $\{click, skip, summarize\}$ on d-nodes, and follow-up summary generation on s-nodes – i.e. a *summGen* edge connecting a d-node to its corresponding s-node. We apply cross-trajectory shuffling and summary-node perturbation operations on this graph to increase training trajectory diversity while maintaining local coherence (see Appendix D).

Trajectory. A *trajectory* τ^{u_j} is a time-ordered path in the UIG for user u_j. It begins at the user's u-node, followed by alternating interactions over d-nodes (documents viewed via clicks/skips) and s-nodes (summaries for documents that were summarized). The path ends in either a d- or s-node. We denote the set of all trajectories by $\mathcal{T}$, with train/test splits $\mathcal{T}_{\text{train}}$ / $\mathcal{T}_{\text{test}}$.

Behavior Triple. We represent each transition as a behavior triple $b_{u_j}^{(t_i)} = \langle hd^{(t_{i-1})}, a^{(t_i)}, tl^{(t_i)} \rangle$ where $hd^{(t_{i-1})}$ is the head node (prior d-node or s-node), $tl^{(t_i)}$ is the tail node (resulting d-node or s-node), and $a^{(t_i)}$ is the action transition from $hd^{(t_{i-1})}$ to $tl^{(t_i)}$ (*click/skip/summarize/summGen*). This structured representation yields a *temporal knowledge graph* of behavior, enabling fine-grained modeling of preference evolution.

Preference Abstraction via Hierarchical UIG. Although the UIG captures rich temporal behavior, long sequences of fine-grained actions can burden sequence encoders. Prior work on sequential modeling shows that adding a *hierarchical abstraction* helps models capture both short- and long-range dependencies more effectively (Xia et al., 2022; Ou et al., 2025; Xue et al., 2022; Zhang et al., 2022). Following this insight, we use a **bi-level** view of the adopted UIG where each behavior triple $b_{u_j}^{(t_i)}$ becomes a **b-node** forming a higher-level trajectory $\tau_b^{u_j}$[1]. This b-tier preserves the temporal order and compresses interactions into interpretable behavior units, making it easier for the model to track preference evolution. Because summaries (s-nodes) are part of the underlying u-tier, strong signals from user-written or model-generated summaries naturally propagate upward into the b-tier, helping the model capture how reading or producing summaries influences future interests–an aspect not addressed in previous work. We construct u- and b-tier graphs from PENS and OpenAI–Reddit. Details are in Appendix C.

Challenges in LLM-based In-Context Personalization. A baseline approach concatenates a user trajectory τ^{u_j} with the query document and prompts an LLM to generate a personalized summary, which is called *In-Context Personalization Learning* (ICPL) (Patel et al., 2024). However, ICPL suffers from major limitations. First, as user

[1]The original fine-grained sequence is the **u-tier**. See Table 9.

history grows, this approach is constrained by context length, and performance can still degrade even when the history is compressed to fit within the limit.Empirically, in long contexts, model performance can be sensitive to the position of relevant information; when key evidence appears in the middle of the prompt, performance degrades, a phenomenon often termed the "lost-in-the-middle" effect (Chen et al., 2025; Liu et al., 2024; Gao et al., 2024). Adding richer user history can therefore degrade personalization quality rather than improve it Patel et al. (2024). Thus, history-injection in ICL is insufficient for robust, evolving personalization.

Problem Reformulation. To overcome these limitations, we reformulate personalized summarization into *three* tasks. **Task-1** predicts the next behavior triple $b_{(q,u_j)}$ for a query document d_q given τ^{u_j}. **Task-2** maps $b_{(q,u_j)}$ and d_q to a top-k keyphrase set. **Task-3** conditions a frozen summarizer on these keyphrases to produce a user-conditioned summary without parameter updates.

4 PerDucer: Personalization Inducer for Summarizers

PerDucer is a *personalized keyphrase inducer* designed to guide frozen summarizers. It operates in two stages –**Task 1** encodes a user's interaction trajectory and predicts the next b-node embedding, and **Task 2** decodes this embedding into top-k personalized keyphrases. We evaluate the keyphrases through downstream summarization rather than as a standalone prediction target. Thus, **Task 3** conditions an external LLM or SLM on the keyphrases to generate the personalized summary. The full workflow is illustrated in Figure 1.

4.1 Next b-Node Prediction (PerDucer Encoder)

Initialization of u-Tier. We first initialize the u-tier trajectory τ^{u_j} by embedding each document (d) and summary (s) node using PromptRank KPE instantiated with a T5-base backbone (220M parameters, 768 hidden size) (Kong et al., 2023). For each behavior triple $b_{u_j}^{(t_i)}$, the head and tail nodes are seeded as $\mathbf{e}_{hd}^{(t_{i-1})}$ and $\mathbf{e}_{tl}^{(t_i)}$. KPE seeding aligns the node embeddings with central themes relevant for keyphrase extraction (see Appendix H.1). The initial user node $\mathbf{e}_{u_j}^{(t_0)}$ uses the title embedding of the first document to mitigate cold start, and actions (*click*, *skip*, *summarize*, *summGen*) are seeded with 4-d one-hot encoding.

Local b-Node Representation (B-RNN). On the b-tier, PerDucer processes the behavior trajectory $\tau_b^{u_j}$ using a stack of b-cells. Each b-cell composes the $b_{u_j}^{(t_i)}$ triplet into a local embedding as $\mathbf{e}_{b_{u_j}}^{(t_i)} = g_{\mathrm{b}}\Big(\mathbf{e}_{hd}^{(t_{i-1})}, \mathbf{e}_{a}^{(t_i)}, \mathbf{e}_{tl}^{(t_i)}\Big)$. The composition function $g_{\mathrm{b}}(\cdot)$ is TransH-inspired (Wang et al., 2014), that learns composition of $hd^{(t_{i-1})}$, $a^{(t_i)}$ and $tl^{(t_i)}$ embeddings (see Appendix E.1). While $\mathbf{e}_{b_{u_j}}$ captures fine-grained behavioral semantics, it remains a *local representation* sensitive to current behavior and near-past historical spans (Figure 2).

History-Aware Encoding via Decay-EMA. Local b-node embeddings reflect only the immediate interactions, but real users exhibit slow preference drift – repeated interest in certain themes and brief detours into others. To capture such longer-range preferences, PerDucer maintains a cumulative, smoothed preference state using a content-aware Decay-based Exponential Moving Average (D-EMA) inspired from MEGA (Ma et al., 2023). At each step t_i, the running snapshot updates as:

$$\mathbf{z}_{b_{u_j}}^{(1:t_i)} = \alpha^{(t_i)} \odot \mathbf{e}_{b_{u_j}}^{(t_i)} + \Big(1 - \tilde{\alpha}^{(t_i)}\Big) \odot \mathbf{z}_{b_{u_j}}^{(t_{1:i-1})} \quad (1)$$

where the gate $\alpha^{(t_i)}$ is a content-aware learnable decay and $\tilde{\alpha}^{(t_i)}$ is a dampening gate acting on past cumulative snapshots $\mathbf{z}^{(t_{1:i-1})}$ (see Appendix E).

Contextualizing Snapshots with FM-Attention (c-MEGA). While D-EMA captures gradual preference drift, users often cyclically return to earlier themes after long gaps. To model such non-local, periodic dependencies, we first contextualize the D-EMA snapshots $\mathbf{z}_{b_{u_j}}^{(1:t_i)}$ using forward-masked self-attention (FM-Attn) as $\mathbf{c}_{b_{u_j}}^{(t_i)} = \mathbf{FM\text{-}Attn}\Big(\mathbf{z}_{b_{u_j}}^{(1:t_i)}\Big)$. We then fuse this contextual signal with the current local behavior embedding $\mathbf{e}_{b_{u_j}}^{(t_i)}$ using a gated residual connection to obtain the **content-aware MEGA (c-MEGA)** state:

$$\mathbf{h}_{b_{u_j}}^{(t_i)} = \gamma^{(t_i)} \odot \mathbf{c}_{b_{u_j}}^{(t_i)} + \Big(1 - \gamma^{(t_i)}\Big) \odot \mathbf{e}_{b_{u_j}}^{(t_i)} \quad (2)$$

where $\gamma^{(t_i)}$ is a learned input gate (full formulation in Appendix E). FM-Attn therefore captures the *cyclical preference* – a hallmark of real-world behavior where themes re-emerge after gaps, while the residual connection of $\mathbf{e}_{b_{u_j}}^{(t_i)}$ recounts the current time-step b-node information. This contextualized state $\mathbf{h}_{b_{u_j}}^{(t_i)}$ serves as the input for predicting the next b-node in the trajectory.

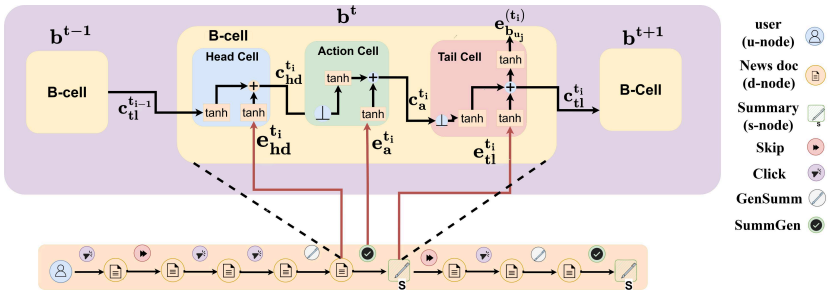

Figure 2: **PerDucer b-tier Encoder**. Each **b-node** is encoded by a **b-cell** with head-, action-, and tail-cells, producing $e_{b_u}^{(t_i)}$ by projecting the head embedding $e_{hd}^{(t_i)}$ with the action embedding $e_a^{(t_i)}$ and injecting the result into the tail embedding $e_{tl}^{(t_i)}$ via another projection.

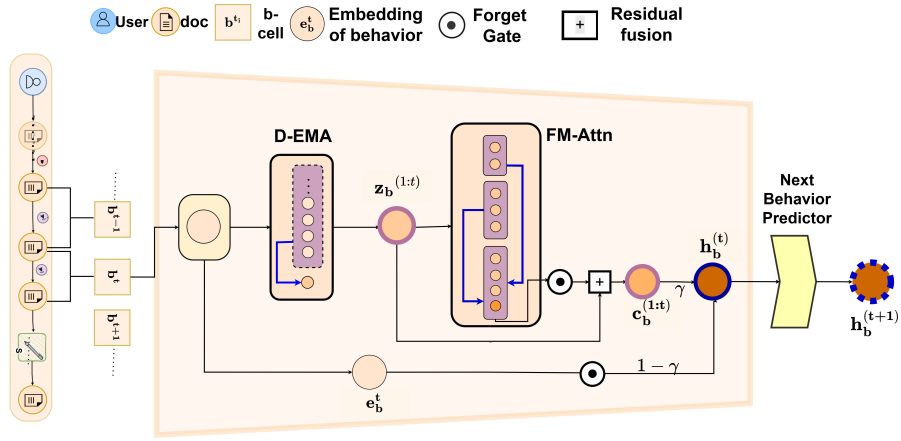

Figure 3: **PerDucer User-Encoder.** The encoder operates on an ordered sequence of b-cells. Each step composes a local b-node embedding and then applies progressive history enrichment via D-EMA and FM-attention to form the c-MEGA state. The final state encodes the behavior history and is mapped to the next b-node embedding.

Predicting the Next b-Node. After processing $\tau_b^{u_j}$ of length l, we use $\mathbf{h}_{b_{u_j}}^{(t_l)}$ as the final encoder state. A prediction head maps this state to the query b-node embedding: $\mathbf{h}_{b_{(\mathrm{q},u_j)}}^{(t_{l+1})} = \mathbf{MLP}\left(\mathbf{h}_{b_{u_j}}^{(t_l)}\right)$. The predicted embedding conditions the keyphrase decoder in Task 2 (See Figure 3 for encoder).

4.2 Personalized Key-Phrase Decoder

Given the predicted query b-node embedding $\mathbf{h}_{b_{(\mathrm{q},u_j)}}^{(t_{l+1})}$ from the PerDucer encoder, the decoder predicts a set of personalized key phrases. We compute a multi-label score over a keyphrase vocabulary $\mathcal{V}_{KP}$: $\hat{\mathcal{P}}_{KP} = MLP(\mathbf{h}_{b_{(\mathrm{q},u_j)}}^{(t_{l+1})}) = \sigma(\mathbf{W}_{KP}\,\phi(\mathbf{h}_{b_{(\mathrm{q},u_j)}}^{(t_{l+1})}))$, where $\hat{\mathcal{P}}_{KP} \in [0,1]^{|\mathcal{V}_{KP}|}$. We construct $\mathcal{V}_{KP}$ by running YAKE (Campos et al., 2020) on PENS and OpenAI-Reddit corpora (size: 176K). The top-k key phrases are selected as $\{kp\}_k = \mathrm{argsort}_k(\hat{\mathcal{P}}_{KP})$ and provide them as conditioning cues to the frozen summarizer in Task 3, enabling PerDucer to act as a behavior-conditioned controller for guiding the summarizer models.

4.3 Guided Personalized Summarization

With Task 2, PerDucer completes its role of generating personalized key phrases that characterize the user's topical intent. In task 3, these key phrases condition frozen summarizers, both LLMs(or SLMs) and vanilla, to produce a user-conditioned summary for the query document d_q.

Guiding LLMs/SLMs. For generative summarizers, we provide the predicted key phrases as structured prompt signals in the form:

***Task**: generate a summary using **key phrases** kp_k for the **query document** d_q. **Conditions**: No hallucinations; keyphrases mandatory.*

This prompt-based injection allows frozen LLMs and SLMs to adapt generation toward the user's inferred preferences without modifying their parameters. We keep the temperature at 0.2, top p

as 0.9, and max tokens as 25 throughout. The full prompt template is reported in Appendix J.

Guiding Vanilla Summarizers. Following Vansh et al. (2023), we adapt standard vanilla summarizers by using the predicted keyphrases as thematic cues. We score each sentence in d_q by its coverage of kp_k. We select the top-m sentences and provide them as an auxiliary input: [**Document Body**, ; **Theme Sentences**].

5 Evaluation

5.1 Training Setup

Training Data. We construct augmented user–interaction graphs (UIGs) from PENS ($\mathcal{T}^{\text{PENS-D}}$) and OpenAI–Reddit ($\mathcal{T}^{\text{OAI}}$) following the procedure in Appendix C. From these, we sample 60K PENS trajectories ($\overline{|d|} = 123$, $\overline{|s|} = 15$) and 35K OAI trajectories ($\overline{|d|} = 37$, $\overline{|s|} = 12$) for training. Each training instance is formed by slicing a user's interaction sequence immediately before a (d, s) pair, so the prefix becomes the user history $\tau_{\text{h}}^{u_j}$ and the next (d, s) serves as (d_q, s_q^*).

Test Data. For PENS evaluation, we create $\mathcal{T}_{\text{test}}^{\text{PENS-D}}$ using each user's stage-1 clicked history and the first 50 stage-2 (d, s) pairs as the fixed prefix $\tau_{\text{h}}^{u_j}$. We then evaluate on a sequence of 150 query documents d_q per user in temporal order. At each step, `PerDucer` predicts keyphrases kp_k for the current query document d_q. A frozen summarizer generates a summary $\hat{s}_q$ conditioned on d_q, and kp_k. We append the pair $(d_q, \hat{s}_q)$ to the history, and use the updated $\tau_h^{u_j}$ for the next query step.

Training `PerDucer`. We train `PerDucer` with a decoder-side key-phrase prediction loss $\mathcal{L}_{\text{KPE}}$ and an encoder-side temporal alignment loss $\mathcal{L}_{\text{ENC}}$. For $\mathcal{L}_{\text{KPE}}$, we extract gold keyphrases (via YAKE) ${kp^*}_{i(1:k)}$ from the reference summary s_q^* to form a multi-hot target over the vocabulary $\mathcal{V}_{KP}$, and apply BCE loss on the predicted relevance distribution $\hat{\mathcal{P}}_{KP}$ as $\mathcal{L}_{\text{KPE}} = -\sum_j (y_j \log \hat{p}_j + (1 - y_j) \log(1 - \hat{p}_j))$. For the encoder, we predict the timestep index of each b-node representation $\mathbf{h}_{b_{u_j}}^{(t_i)}$. A learnable position head W_{pos} maps each b-node to a temporal distribution $\hat{\mathcal{P}}_{\text{pos}} = \text{SoftMax}(W_{\text{pos}} \mathbf{h}_{b_{u_j}}^{(t_i)})$, and we optimize with mean NLL loss as $\mathcal{L}_{\text{ENC}} = -\sum_{i=1}^{l} \log \hat{p}(t_i)$. The final objective is $\mathcal{L}_{\texttt{PerDucer}} = 0.5 \cdot \mathcal{L}_{\text{KPE}} + 0.5 \cdot \mathcal{L}_{\text{ENC}}$.

5.2 Baseline Summarization Models

LLMs. For RQ-1, we evaluate four frozen LLMs–Mistral-7B-Instruct (Jiang et al., 2023), DeepSeek-R1-Distill-Qwen-14B, LLaMA-2-13B-Chat, and Zephyr-7B-β. We use the 2-shot prompts from Patel et al. (2024), and prompt chaining where applicable. Our goal is not to identify the "best" LLM, but to demonstrate that `PerDucer` consistently improves personalization *keeping the underlying summarizer frozen*. This makes `PerDucer` a model-agnostic and resource-efficient (see Table 10).

Non-personalized Summarizers with Cue Injection (Oracle). To estimate an upper bound for cue-driven personalization (RQ-1b), we include two strong generic summarizers: BigBird-Pegasus (Zaheer et al., 2020) and SimCLS (Liu and Liu, 2021). Following Vansh et al. (2023), we augment the input with gold keyphrase cues, yielding an oracle-style reference for how much personalization can be induced in non-personalized systems.

Small Language Models. For RQ-2, we assess two frozen SOTA SLMs–Qwen2.5-0.5B-Instruct (Qwen et al., 2025) and SmolLM2-1.7B-Instruct (Allal et al., 2025) for examining whether `PerDucer` can lift smaller models toward LLM-level personalization quality.

Personalized Summarizer Baselines. For RQ-3, we benchmark three personalized frameworks: PENS (Ao et al., 2021), GTP (Song et al., 2023), and Signature-Phrase (Cai et al., 2023). PENS incorporates external user encoders such as NRMS (Wu et al., 2019b), NAML (Wu et al., 2019a), and EBNR (Okura et al., 2017). GTP integrates its Transformer-based TrRMIo encoder. Signature-Phrase models user-specific keyphrases. The diverse architectures make them relevant for comparison. All personalized baselines are finetuned for 2 epochs under their reported *training regimes*.

5.3 Evaluation Metrics

Personalization Quality. To evaluate how effectively `PerDucer` improves personalization, we use `PerSEval` (PSE), the metric proposed by Dasgupta et al. (2024). `PerSEval` jointly measures (i) alignment with user-specific expected summaries and (ii) preservation of factual accuracy. We report PSE-JSD, PSE-SU4, and PSE-METEOR as primary personalization metrics.

Content Accuracy. For completeness and content fidelity assessment, we also report standard content-overlap metrics, ROUGE-SU4 (Lin, 2004),

Model	2-shot			B-RNN			D-EMA			c-EMA			c-MEGA (PerDucer)		
	JSD	SU4	METEOR	JSD	SU4	METEOR	JSD	SU4	METEOR	JSD	SU4	METEOR	JSD	SU4	METEOR
Mistral-7B	0.226	0.086	0.083	0.211	0.089	0.097	0.313	0.103	0.307	0.317	0.118	0.311	0.342	0.129	0.316
DeepSeek-R1	0.238	0.087	0.102	0.215	0.092	0.108	0.321	0.114	0.311	0.325	0.124	0.316	0.348	0.134	0.324
Zephyr-7B-β	0.231	0.078	0.079	0.205	0.088	0.103	0.319	0.111	0.309	0.326	0.118	0.314	0.344	0.131	0.319
LLaMA-13B	0.217	0.071	0.078	0.206	0.093	0.105	0.321	0.113	0.306	0.329	0.117	0.313	0.337	0.128	0.321
Qwen2.5-0.5B	NA*	NA*	NA*	0.202	0.089	0.101	0.296	0.107	0.284	0.325	0.114	0.307	0.327	0.124	0.315
smolLM2-1.7B	NA*	NA*	NA*	0.211	0.093	0.198	0.305	0.113	0.297	0.326	0.116	0.314	0.343	0.128	0.322

Table 1: **RQ-1/2: Personalization boost consistency across LLMs & SLMs via PerDucer**. *Obs-1:* While D-EMA & c-EMA show consistent gains, c-MEGA yields the strongest gains overall; *Obs-2:* SLMs remain competitive under constrained personalization ($p \leq 0.05$; NA*: SLMs are not prompted with histories due to limited context).

ROUGE-L (Lin and Och, 2004), and BLEU (Papineni et al., 2002), computed against gold summaries. We additionally report the semantic similarity–based metric BERTScore (Zhang et al., 2020).

Human-Preference Alignment. We employ a survey-based human evaluation and a proxy human judgment (HJ) measure. In the survey-based setting, annotators rate the semantic closeness of model outputs to gold references (scale 1-6, higher is better) without model identity disclosure (Figure 5). For pseudo HJ, we use the multi-domain OpenAI–Reddit dataset containing human ratings for summaries from nine models, selecting each user's highest-rated summary (rating 7) as a *surrogate human reference*. We compute RMSD divergence between model outputs and this reference, and interpolate approximate HJ scores using the empirical rating–RMSD mapping (Table 16).

6 Results & Observations

6.1 RQ-1: Boosting LLM Personalization

We evaluate whether PerDucer improves the personalization for frozen LLM summarizers relative to strong prompt-based baselines (Section 5.2). All LLMs are evaluated under consistent prompt settings, as reported in Section 4.3.

Across all four LLMs, PerDucer yields consistent and substantial gains in personalization. PerDucer-boosted LLMs improve over 2-shot prompting by **+0.115/+0.051/+0.235** on PSE-JSD/SU4/METEOR, respectively (Table 1). Relative to prompt chaining, DeepSeek improves by **+0.27/+0.106/+0.297**. These improvements support keyphrase-guided conditioning as an effective alternative to history prompting (Table 12).

Importantly, personalization gains do not come at the expense of content fidelity. PerDucer+DeepSeek improves over the strongest baseline LLM, DeepSeek-14B, by **+23.9** (SU4), **+14.5** (RL), **+4.96** (BLEU), and **+1.83** (BScore) (Table 2), indicating alignment with user preferences rather than metric artifacts. PerDucer-boosted LLMs also achieve the maximum proxy human rating (7/7) under the RMSD-based human-judgment interpolation scheme (Table 16).

Model	SU4	RL	BLEU	BScore	HJ (survey/interpolated)
PENS-NRMS-T2	13.64	21.03	4.48	86.13	3.04/2
GTP-TrRMIo	21.91	28.31	10.31	88.53	2.92/2
SP-Individual	19.54	25.18	8.90	86.61	2.68/3
DeepSeek-14B (2-shot)	19.57	29.72	12.68	89.43	3.12/5
LLaMA-2 (2-shot)	18.31	29.54	11.85	88.76	3.36/5
PerDucer+DeepSeek14B	**34.05**	**53.83**	**17.64**	**91.16**	**3.47/7**
PerDucer+Zephyr-β	*32.55*	*51.46*	*16.48*	*90.84*	3.38/**7**

Table 2: **Personalized Summarization Performance (Accuracy & Human Judgment).** Standard accuracy metrics & Average human ratings w.r.t. gold-references (Higher is better across all metrics ($p \leq 0.05$)).

Inducing Personalization in Vanilla Summarizers (Oracle Approximation). To isolate the effectiveness of PerDucer's personalized keyphrase induction (Task 2), we evaluate non-personalized summarizers augmented with PerDucer-generated cues. We compare them against oracle versions that use gold-reference summaries as cues (Section 5.2). This comparison measures how closely PerDucer-generated keyphrases can approximate oracle cue injection under the same summarization setup. PerDucer consistently lifts the baseline vanilla summarizers towards their oracle performance. BigBird-Pegasus achieves **80.4%/77.6%/78.7%** of oracle performance on PSE-JSD, PSE-SU4, and PSE-METEOR, respectively (Table 5). These results indicate that PerDucer learns highly effective personalized keyphrases that capture most of the user-specific signal available in gold cues.

6.2 RQ-2: Boosting Small Language Models

Recent studies show that small language models (SLMs) can approach LLM performance on well-structured tasks (Fu et al., 2024; Xu et al., 2025). Since PerDucer reframes personalized summarization as keyphrase-guided conditioning, we evaluate whether SLMs benefit from the same mechanism.

We observe that PerDucer substantially improves both SLMs (see Table 1). SmolLM2-1.7B-Instruct differs from DeepSeek by a marginal **0.004** on average across PSE-JSD, PSE-SU4, and PSE-METEOR. Qwen2.5-0.5B-Instruct also benefits from PerDucer, trailing LLMs by only **0.01/0.007/0.005** on PSE-JSD/SU4/METEOR. This validates that keyphrase-guided conditioning enables compact models to approximate LLM-level personalization despite their limited context windows and lack of effective in-context learning.

Ablation: Effectiveness of c-MEGA. We ablate the encoder variants: (i) vanilla b-tier, (ii) b-tier + D-EMA, (iii) D-EMA + FM-Attn (c-EMA), and (iv) full c-MEGA. As shown in Table 1, c-MEGA consistently outperforms the strongest baseline (c-EMA) by **0.015/0.011/0.007**↑ on PSE-JSD, PSE-SU4, and PSE-METEOR, respectively. This highlights the benefit of jointly modeling long-term preference drift and non-local dependencies.

Ablation: Number of Keyphrases and Temperature. We vary the number of extracted keyphrases $k \in \{5, 10, 15\}$ (gold average: 20.23). Performance peaks at $k = 10$ (Table 3). Averaged across PSE-JSD, PSE-SU4, and PSE-METEOR, $k = 10$ improves over $k = 5$ by **0.209**↑ and over $k = 15$ by **0.023**↑. Token-level analysis reports 66% coverage and ROUGE-1 recall of 0.65. We examine the faithfulness of the default temperature (T=0.2) relative to higher temperatures. Increasing T consistently degrades personalization quality across PSE metrics, with an average degradation of 0.038 at T=0.5 and 0.086 at T=0.8 (Table 13).

Cross-Domain Generalization. We train PerDucer on OpenAI–Reddit $\mathcal{T}^{\text{OAI}}_{\text{train}}$ (29 non-news domains) and evaluate on $\mathcal{T}^{\text{OAI}}_{\text{test}}$. c-MEGA consistently improves personalization across LLMs (best Zephyr-β: **0.102/0.036/0.101**↑ on PSE-JSD/SU4/METEOR). Without retraining, the same model also improves performance on $\mathcal{T}^{\text{PENS-D}}_{\text{test}}$ by **0.025/0.007/0.052**↑, demonstrating robustness under cross-domain shift (Table 4).

Ablation: History Rollout-Length Stability. We evaluate the first $n_q \in \{50, 100, 150\}$ query d_q, which increases $\tau_{\text{h}}^{u_j}$ with $(d_q, \hat{s}_q)$ pairs. DeepSeek 2-shot shows an average drop of 12.1% from $n_q = 50$ to $n_q = 150$, and Mistral 2-shot drops by 13.6% (PSE-JSD/SU4/METEOR averaged). PerDucer+DeepSeek remains stable with a 1.9% average drop from $n_q = 50$ to $n_q = 150$, while PerDucer+Mistral remains stable with a 0.6% average increase (see Table 14).

Stress Test: Sparse Click-only Histories. We project each PENS test trajectory to a click-only history by removing *skip* and *summarize* interactions. Under this projection, 2-shot baselines degrade sharply, with DeepSeek (2-shot baseline) dropping by 40.7/31.9/26.8% from actual performance. PerDucer+DeepSeek shows much lesser degradation (16.7/20.2/15.2%), indicating the robustness of PerDucer as a personalized keyphrase extractor and personalization inducer (Table 15).

Ablation: Oracle Rollout as Upperbound. We replace the history update $\hat{s}_q$ by s^*_q when appending s-nodes into $\tau_{\text{h}}^{u_j}$ to remove self-conditioning error accumulation. Relative to autoregressive rollout, rolling out s^*_q improves PSE-JSD/SU4/METEOR by **+0.36/+0.37/+0.18** (Table 12).

6.3 RQ-3: Boosted Vanilla vs. Specialized Personalized Summarizers

We further evaluate PerDucer as a personalization booster by comparing PerDucer-augmented *frozen* vanilla summarizers against state-of-the-art *specialized* personalized models. Despite not being fine-tuned for personalization, PerDucer-boosted BigBird-Pegasus exceeds GTP by **0.18/0.094/0.113** on PSE-JSD, PSE-SU4, and PSE-METEOR (Table 5). These results indicate that inducing personalization via keyphrase-guided control is more effective than directly modeling long user histories with self-attention or RNN-based architectures, as adopted by GTP and PENS.

7 Conclusion

Personalized summarization remains challenging due to long, mixed user histories that combine positive and negative signals and are difficult for LLMs to encode in-context. We introduce PerDucer that reframes personalization as **keyphrase-guided summarization** where user interaction histories are compressed into personalized keyphrases that guide frozen summarizers. Across models and settings, PerDucer delivers consistent personalization gains (averaging **0.18**↑), including strong improvements for LLMs and compact models that struggle with direct history injection. Across our evaluated settings, these results suggest that personalization via lightweight control signals can be an effective and efficient alternative to direct history injection.

LLM	$k = 5$ Keyphrases			$k = 10$ Keyphrases			$k = 15$ Keyphrases		
	PSE-JSD	PSE-SU4	PSE-MET	PSE-JSD	PSE-SU4	PSE-MET	PSE-JSD	PSE-SU4	PSE-MET
Mistral-7B	0.075	0.045	0.052	0.342	0.129	0.316	0.312	0.118	0.287
DeepSeek-R1	0.077	0.048	0.055	**0.348**	**0.134**	**0.324**	0.318	0.123	0.295
Zephyr-7B-β	0.066	0.044	0.051	0.344	0.131	0.319	0.314	0.120	0.291
LLaMA-13B	0.065	0.043	0.039	0.337	0.128	0.321	0.308	0.117	0.292
Qwen2.5-0.5B	0.063	0.037	0.039	0.327	0.124	0.315	0.300	0.114	0.287
smolLM2-1.7B	0.068	0.047	0.054	0.343	0.128	0.322	0.313	0.118	0.294

Table 3: **Top-k keyphrase ablation.** Using k=10 keyphrases consistently yields the best PSE performance across LLMs ($p \leq 0.05$).

Model	w/ history			PENS Test			OpenAI Test		
	JSD	SU4	METEOR	JSD	SU4	METEOR	JSD	SU4	METEOR
DeepSeek-R1	0.243	0.095	0.109	0.262	0.099	0.244	**0.320**	**0.126**	**0.292**
Zephyr-7B-β	0.214	0.087	0.104	0.245	0.092	0.208	0.316	0.123	0.289
LLaMA-13B	0.232	0.093	0.107	0.257	**0.100**	**0.252**	0.319	0.124	0.291
Mistral-7B	0.226	0.088	0.103	0.246	0.091	0.235	0.309	0.114	0.284
smolLM2-1.7B	NA*	NA*	NA*	**0.262**	0.096	0.239	0.321	0.121	0.289
Qwen2.5-0.5B	NA*	NA*	NA*	0.242	0.085	0.228	0.297	0.108	0.257

Table 4: **Cross-domain generalizability under domain shift.** `PerDucer` trained on OpenAI–Reddit and evaluated on PENS and OpenAI (All reported metrics are PSE variants). *Obs-1:* Consistent improvement over 2-shot baselines via `PerDucer` on OpenAI-Reddit indicates strong generalizability. *Obs-2:* Scores on PENS remain competitive with respect to `PerDucer` trained on PENS, demonstrating effective transferability.

Type	Model	PSE-JSD	PSE-SU4	PSE-METEOR
Specialized Models	PENS-NAML-T1	0.021	0.014	0.016
	PENS-EBNR-T1	0.015	0.010	0.011
	PENS-EBNR-T2	0.011	0.008	0.009
	PENS-NRMS-T1	0.015	0.011	0.011
	PENS-NRMS-T2	0.008	0.007	0.007
	GTP	0.024	0.017	0.019
	SP-Individual	0.017	0.015	0.014
Generic (+ Title Oracle)	BigBirdPegasus	***0.253***	***0.143***	***0.168***
	SimCLS	0.157	0.032	0.016
Generic (+ `PerDucer` KP)	BigBirdPegasus	**0.204**	**0.111**	**0.132**
	SimCLS	0.085	0.023	0.013

Table 5: **RQ-3: Performance of vanilla summarizers compared against specialized personalized models on PENS**. *Obs-1:* Boosted vanilla models outperform all specialized SOTA personalized summarizers. *Obs-2:* `PerDucer`-guided keyphrases boost vanilla models toward oracle-level personalization. ($p \leq 0.05$).

8 Limitations & Future Scope

We propose `PerDucer` as a lightweight framework that infers keyphrases as preference signals and injects them into frozen summarization models. This design enables efficient personalization without modifying the backbone model and represents preferences at an abstract level. We are actively extending this abstraction toward more explicit action-level and structured control representations. Incorporating mechanisms for modeling preference evolution, intervention, and controllable adaptation over time is a promising direction for future work. We instantiate `PerDucer` for text summarization in this paper. Extending the framework to broader domains and generation settings, such as multi-turn dialogue and sequential recommendation, remains an important step toward assessing the generality of the proposed control mechanism.

9 Acknowledgement

Tanmoy Chakraborty acknowledges the support of the Anusandhan National Research Foundation (DST/INT/USA/NSF-DST/Tanmoy/P2/2024), Google GCP Grant, and Rajiv Khemani Young Faculty Chair Professorship in Artificial Intelligence.

References

Loubna Ben Allal, Anton Lozhkov, Elie Bakouch, Gabriel Martín Blázquez, Guilherme Penedo, Lewis Tunstall, Andrés Marafioti, Hynek Kydlíček, Agustín Piqueres Lajarín, Vaibhav Srivastav, Joshua Lochner, Caleb Fahlgren, Xuan-Son Nguyen, Clémentine Fourrier, Ben Burtenshaw, Hugo Larcher, Haojun Zhao, Cyril Zakka, Mathieu Morlon, and 3 others. 2025. Smollm2: When smol goes big – data-centric training of a small language model. *Preprint*, arXiv:2502.02737.

Xiang Ao, Xiting Wang, Ling Luo, Ying Qiao, Qing He, and Xing Xie. 2021. PENS: A dataset and generic framework for personalized news headline generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 82–92, Online. Association for Computational Linguistics.

Satanjeev Banerjee and Alon Lavie. 2005. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 65–72, Ann Arbor, Michigan. Association for Computational Linguistics.

Pengshan Cai, Kaiqiang Song, Sangwoo Cho, Hongwei Wang, Xiaoyang Wang, Hong Yu, Fei Liu, and Dong

Yu. 2023. Generating user-engaging news headlines. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3265–3280, Toronto, Canada. Association for Computational Linguistics.

Ricardo Campos, Vítor Mangaravite, Arian Pasquali, Alípio Jorge, Célia Nunes, and Adam Jatowt. 2020. Yake! keyword extraction from single documents using multiple local features. *Inf. Sci.*, 509(C):257–289.

Parthiv Chatterjee, Shivam Sonawane, Amey Hengle, Aditya Tanna, Sourish Dasgupta, and Tanmoy Chakraborty. 2025. Diversity augmentation of dynamic user preference data for boosting personalized text summarizers. *Preprint*, arXiv:2510.10082.

Yinpeng Chen, DeLesley Hutchins, Aren Jansen, Andrey Zhmoginov, David Racz, and Jesper Sparre Andersen. 2025. MELODI: Exploring memory compression for long contexts. In *The Thirteenth International Conference on Learning Representations*.

Sourish Dasgupta, Ankush Chander, Tanmoy Chakraborty, Parth Borad, and Isha Motiyani. 2024. PerSEval: Assessing personalization in text summarizers. *Transactions on Machine Learning Research*.

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, and 8 others. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *Preprint*, arXiv:2501.12948.

Zi-Yi Dou, Pengfei Liu, Hiroaki Hayashi, Zhengbao Jiang, and Graham Neubig. 2021a. GSum: A general framework for guided neural abstractive summarization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4830–4842, Online. Association for Computational Linguistics.

Zi-Yi Dou, Pengfei Liu, Hiroaki Hayashi, Zhengbao Jiang, and Graham Neubig. 2021b. GSum: A general framework for guided neural abstractive summarization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4830–4842, Online. Association for Computational Linguistics.

Alexander Fabbri, Irene Li, Tianwei She, Suyi Li, and Dragomir Radev. 2019. Multi-news: A large-scale multi-document summarization dataset and abstractive hierarchical model. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1074–1084, Florence, Italy. Association for Computational Linguistics.

Xue-Yong Fu, Md Tahmid Rahman Laskar, Elena Khasanova, Cheng Chen, and Shashi Tn. 2024. Tiny titans: Can smaller large language models punch above their weight in the real world for meeting summarization? In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 6: Industry Track)*, pages 387–394, Mexico City, Mexico. Association for Computational Linguistics.

Muhan Gao, TaiMing Lu, Kuai Yu, Adam Byerly, and Daniel Khashabi. 2024. Insights into LLM long-context failures: When transformers know but don't tell. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 7611–7625, Miami, Florida, USA. Association for Computational Linguistics.

Junxian He, Wojciech Kryscinski, Bryan McCann, Nazneen Rajani, and Caiming Xiong. 2022a. CTRLsum: Towards generic controllable text summarization. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Junxian He, Wojciech Kryscinski, Bryan McCann, Nazneen Rajani, and Caiming Xiong. 2022b. CTRLsum: Towards generic controllable text summarization. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 5879–5915, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Karl Moritz Hermann, Tomáš Kočiský, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'15, page 1693–1701, Cambridge, MA, USA. MIT Press.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b. *Preprint*, arXiv:2310.06825.

Frederic Kirstein, Terry Ruas, Robert Kratel, and Bela Gipp. 2024. Tell me what I need to know: Exploring LLM-based (personalized) abstractive multi-source meeting summarization. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pages 920–939, Miami, Florida, US. Association for Computational Linguistics.

Aobo Kong, Shiwan Zhao, Hao Chen, Qicheng Li, Yong Qin, Ruiqi Sun, and Xiaoyan Bai. 2023. PromptRank: Unsupervised keyphrase extraction using prompt. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics*

(Volume 1: Long Papers), pages 9788–9801, Toronto, Canada. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880. Association for Computational Linguistics.

Zekun Li, Baolin Peng, Pengcheng He, Michel Galley, Jianfeng Gao, and Xifeng Yan. 2023. Guiding large language models via directional stimulus prompting. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, NIPS '23, Red Hook, NY, USA. Curran Associates Inc.

Junhong Lian, Xiang Ao, Xinyu Liu, Yang Liu, and Qing He. 2025. Panoramic interests: Stylistic-content aware personalized headline generation. In *Companion Proceedings of the ACM on Web Conference 2025*.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Chin-Yew Lin and Franz Josef Och. 2004. Automatic evaluation of machine translation quality using longest common subsequence and skip-bigram statistics. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*, pages 605–612.

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

Yixin Liu and Pengfei Liu. 2021. SimCLS: A simple framework for contrastive learning of abstractive summarization. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 1065–1072. Association for Computational Linguistics.

Xuezhe Ma, Chunting Zhou, Xiang Kong, Junxian He, Liangke Gui, Graham Neubig, Jonathan May, and Luke Zettlemoyer. 2023. Mega: Moving average equipped gated attention. In *The Eleventh International Conference on Learning Representations*.

M.L. Menéndez, J.A. Pardo, L. Pardo, and M.C. Pardo. 1997. The jensen-shannon divergence. *Journal of the Franklin Institute*, 334(2):307–318.

Shumpei Okura, Yukihiro Tagami, Shingo Ono, and Akira Tajima. 2017. Embedding-based news recommendation for millions of users. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '17, page 1933–1942, New York, NY, USA. Association for Computing Machinery.

Zhonghong Ou, Xiao Zhang, and Zhu. 2025. Ls-tgnn: Long and short-term temporal graph neural network for session-based recommendation. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Divya Patel, Pathik Patel, Ankush Chander, Sourish Dasgupta, and Tanmoy Chakraborty. 2024. Are large language models in-context personalized summarizers? get an iCOPERNICUS test done! In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 16820–16842, Miami, Florida, USA. Association for Computational Linguistics.

Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, and 25 others. 2025. Qwen2.5 technical report. *Preprint*, arXiv:2412.15115.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Yun-Zhu Song, Yi-Syuan Chen, Lu Wang, and Hong-Han Shuai. 2023. General then personal: Decoupling and pre-training for personalized headline generation. *Transactions of the Association for Computational Linguistics*, 11:1588–1607.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, and 1 others. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv e-prints*, pages arXiv–2307.

Lewis Tunstall, Edward Beeching, Nathan Lambert, Nazneen Rajani, Kashif Rasul, Younes Belkada, Shengyi Huang, Leandro von Werra, Clémentine Fourrier, Nathan Habib, and 1 others. 2023. Zephyr: Direct distillation of lm alignment. *arXiv e-prints*, pages arXiv–2310.

Rahul Vansh, Darsh Rank, Sourish Dasgupta, and Tanmoy Chakraborty. 2023. Accuracy is not enough: Evaluating personalization in summarizers. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 2582–2595, Singapore. Association for Computational Linguistics.

Michael Völske, Martin Potthast, Shahbaz Syed, and Benno Stein. 2017. TL;DR: Mining Reddit to learn automatic summarization. In *Proceedings of the Workshop on New Frontiers in Summarization*, pages 59–63, Copenhagen, Denmark. Association for Computational Linguistics.

Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge graph embedding by translating on hyperplanes. *Proceedings of the AAAI Conference on Artificial Intelligence*, 28(1).

Chuhan Wu, Fangzhao Wu, Mingxiao An, Jianqiang Huang, Yongfeng Huang, and Xing Xie. 2019a. Neural news recommendation with attentive multi-view learning. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*. International Joint Conferences on Artificial Intelligence Organization.

Chuhan Wu, Fangzhao Wu, Suyu Ge, Tao Qi, Yongfeng Huang, and Xing Xie. 2019b. Neural news recommendation with multi-head self-attention. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6389–6394, Hong Kong, China. Association for Computational Linguistics.

Fangzhao Wu, Ying Qiao, Jiun-Hung Chen, Chuhan Wu, Tao Qi, Jianxun Lian, Danyang Liu, Xing Xie, Jianfeng Gao, Winnie Wu, and Ming Zhou. 2020. MIND: A large-scale dataset for news recommendation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3597–3606, Online. Association for Computational Linguistics.

Lianghao Xia, Chao Huang, Yong Xu, and Jian Pei. 2022. Multi-behavior sequential recommendation with temporal graph transformer. *IEEE Transactions on Knowledge and Data Engineering*, 35(6):6099–6112.

Wen Xiao, Yujia Xie, Giuseppe Carenini, and Pengcheng He. 2024. Personalized abstractive summarization by tri-agent generation pipeline. In *Findings of the Association for Computational Linguistics: EACL 2024*, pages 570–581, St. Julian's, Malta. Association for Computational Linguistics.

Borui Xu, Yao Chen, Zeyi Wen, Weiguo Liu, and Bingsheng He. 2025. Evaluating small language models for news summarization: Implications and factors influencing performance. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 4909–4922, Albuquerque, New Mexico. Association for Computational Linguistics.

Lyuxin Xue, Deqing Yang, and Yanghua Xiao. 2022. Factorial user modeling with hierarchical graph neural network for enhanced sequential recommendation. In *2022 IEEE international conference on multimedia and expo (ICME)*, pages 01–06. IEEE.

Zhao Yang, Junhong Lian, and Xiang Ao. 2023. Fact-preserved personalized news headline generation. *2023 IEEE International Conference on Data Mining (ICDM)*.

Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, and Amr Ahmed. 2020. Big bird: Transformers for longer sequences. In *Advances in Neural Information Processing Systems*, volume 33, pages 17283–17297.

Lemei Zhang, Peng Liu, Marcus Tiedemann Oekland Henriksboe, Even W. Lauvrak, Jon Atle Gulla, and Heri Ramampiaro. 2024. Personalsum: A user-subjective guided personalized summarization dataset for large language models. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Qi Zhang, Bin Wu, Zhongchuan Sun, and Yangdong Ye. 2022. Gating augmented capsule network for sequential recommendation. *Knowledge-Based Systems*, 247:108817.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.

A Measuring Degree-of-Personalization

A.1 Motivation

Vansh et al. (2023) proposed `EGISES` to quantify a model's *insensitivity to subjectivity*, i.e., lack of personalization, via relative deviation rather than absolute quality. Building on this, Dasgupta et al. (2024) defined the *Degree-of-Responsiveness* (`DEGRESS`), which measures how closely a model's inter-user variation aligns with variation in user expectations.

A.2 `DEGRESS` Formulation

Summary-level `DEGRESS`. Given document d_i and user profile u_{ij}, the summary-level responsiveness of model $M_{\boldsymbol{\theta},u}$ is:

$$\texttt{DEGRESS}(s_{u_{ij}}|(d_i,u_{ij})) = \frac{1}{|\mathbf{U}_{d_i}|}\sum_k \frac{\min(X_{ijk},Y_{ijk})+\epsilon}{\max(X_{ijk},Y_{ijk})+\epsilon}, \tag{3}$$

where $X_{ijk} = \sigma(u_{ij},u_{ik})\cdot w^u_{ijk}$,
$Y_{ijk} = \sigma(s_{u_{ij}},s_{u_{ik}})\cdot w^s_{ijk}$.

Here, $\sigma(\cdot,\cdot)$ denotes a summary distance metric, and w^u, w^s are softmax-normalized weights based on divergence from the source document d_i. Lower $\texttt{DEGRESS}(s_u|(d,u))$ indicates weaker responsiveness.

System-level DEGRESS.

$$\texttt{DEGRESS}(M_{\boldsymbol{\theta},u}) = \frac{1}{|\mathbf{D}|}\sum_i \frac{1}{|\mathbf{U}_{d_i}|}\sum_j \texttt{DEGRESS}(s_{u_{ij}}|(d_i,u_{ij})). \tag{4}$$

A.3 PerSEval: Accuracy-Aware Responsiveness

While DEGRESS ignores accuracy degradation, PerSEval introduces an *Effective* DEGRESS *Penalty* (EDP) to penalize poor faithfulness without overwhelming responsiveness.

Summary-level PerSEval.

$$\texttt{PerSEval}(s_{u_{ij}}|(d_i,u_{ij})) = \texttt{DEGRESS}(s_{u_{ij}}|(d_i,u_{ij}))\cdot\texttt{EDP}_{ij}, \tag{5}$$

with

$$\texttt{EDP}_{ij} = 1 - \frac{1}{1+10^{\alpha}\exp(-10^{\beta}\cdot dp_{ij})}, \tag{6}$$

where $dp_{ij} = \texttt{ADP}_i + \texttt{ACP}_{ij}$.

Accuracy penalties. The **Accuracy Drop Penalty** (ADP) captures worst-case accuracy:

$$\texttt{ADP}_i = \frac{1}{1+10^{\gamma}\exp\left(-10\frac{\sigma^*(s_{u_i},u_i)|d}{1-\sigma^*(s_{u_i},u_i)|d+\epsilon}\right)}, \tag{7}$$

where $\sigma^*(s_{u_i},u_i)|d = \min_j \sigma(s_{u_{ij}},u_{ij})|d_i$.

The **Accuracy Consistency Penalty** (ACP) penalizes inconsistency across users:

$$\texttt{ACP}_{ij} = \frac{1}{1+10^{\gamma}\exp\left(-10\frac{\sigma(s_{u_{ij}},u_{ij})|d_i-\sigma^*(s_{u_i},u_i)|d}{\overline{\sigma}(s_{u_i},u_i)|d-\sigma^*(s_{u_i},u_i)|d+\epsilon}\right)}, \tag{8}$$

with $\overline{\sigma}(s_{u_i},u_i)|d = \frac{1}{|\mathbf{U}_{d_i}|}\sum_j \sigma(s_{u_{ij}},u_{ij})|d_i$.

System-level PerSEval.

$$\texttt{PerSEval}(M_{\boldsymbol{\theta},u}) = \frac{1}{|\mathbf{D}|}\sum_i \frac{1}{|\mathbf{U}_{d_i}|}\sum_j \texttt{PerSEval}(s_{u_{ij}}|(d_i,u_{ij})) \tag{9}$$

A.4 PSE Metric Variants

PSE-SU4. Uses ROUGE-SU4 F1 as similarity, i.e., $\sigma_{\text{SU4}}(G,R) = 1 - \text{ROUGE-SU4}$ to correlate well with human judgment (Dasgupta et al., 2024).

PSE-JSD. Uses JSD (Menéndez et al., 1997) as divergence metric as:

$$\sigma_{\text{JSD}}(G,R) = \tfrac{1}{2}D_{\text{KL}}(P_G\|M) + \tfrac{1}{2}D_{\text{KL}}(P_R\|M), \quad M = \tfrac{1}{2}(P_G+P_R). \tag{10}$$

PSE-Meteor. Uses METEOR similarity converted to distance: $\sigma_{\text{Meteor}}(G,R) = 1 - \text{METEOR}(G,R)$, where METEOR aligns unigrams with stemming, synonymy, and fragmentation penalties (Banerjee and Lavie, 2005).

B Datasets

B.1 PENS Dataset

The PENS dataset (Ao et al., 2021) includes 113,762 news articles across 15 topics. Each article contains an ID, title (avg. 10.5 words), body (avg. 549 words), and category, with titles linked to WikiData entities. The dataset also includes user interaction data, such as impressions and click behaviors, combined with news bodies and headlines from the MIND dataset (Wu et al., 2020)

PENS training set. For training, 500k user-news impressions were sampled from June 13 to July 3, 2019. Each log records user interaction as [uID, tmp, clkNews, uclkNews, clkedHis], where 'clkNews' and 'uclkNews' represent clicked and unclicked news, and 'clkedHis' refers to the user's prior clicked articles, sorted by click time. The training data for PerDucer, as discussed in Section 5.1, shows high preference shift. This inherently supports that personalizing UX is strongly dependent on the temporal dynamics of the user. The stats are in the table 8.

PENS test set. To create an offline testbed, 103 English-speaking students reviewed 1,000 headlines in stage-1, and then selected 50 articles, and created preferred headlines (i.e., expected gold-reference summaries) for 200 unseen articles in stage-2 (see Figure 4). Each article was reviewed by four participants. Editors checked for factual accuracy, discarding incorrect headlines. The high-quality remaining headlines serve as personalized gold-standard references in the PENS dataset.

B.2 OpenAI (Reddit) Dataset

The OpenAI (Reddit) dataset (Völske et al., 2017) comprises 123,169 Reddit posts collected from 29 distinct subreddits. This dataset provides both OpenAI-generated and human-written summaries and is organized into two splits: Comparisons, used for training and validation, and Axis, designated for validation and testing. A curated subset of 1,038 posts was processed by 13 different summarization policies, resulting in the generation of 7,713 summaries. These summaries underwent evaluation by 64 annotators who rated paired summaries

Category	Dimension	Value
	Article Statistics	
General Stats	# Topics	15
	# Articles	113,762
	Avg. Title Length	10.5 words
	Avg. Body Length	549 words
	Training Dataset Statistics	
Interaction Data	# User–News Impressions (anon.)	500,000
	# Users (anon.)	445,000
	Time Period	June 13–July 3, 2019
	Interaction Fields	[uID, tmp, clkNews, uclkNews, clkedHis]
	Test Dataset Statistics	
Participants	# Participants	103
	Participant Category	English-speaking college students
	# Articles	3,940
	Browsed Headlines (Click + Skip)	1,000 / participant
	Min. Interested (Click) Headlines	50 / participant
Gold Reference	Summarized Article Bodies	200 / participant
	Avg. Summaries per Article	4

Table 6: **MS/CAS PENS dataset statistics.** Summary of article corpus, user interaction logs, and human-written personalized headline annotations.

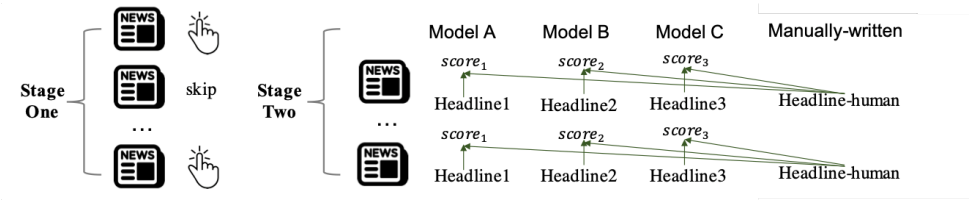

Figure 4: **PENS test data creation (Ao et al., 2021).**

based on selection preferences, confidence in their ratings, and dimensions such as accuracy, coherence, coverage, and overall quality. Notably, unlike datasets like PENS, these summaries are not linked to individual annotators or their reading histories, which means they lack elements of personalization and contextual user information.

C UIG Construction from Preference Datasets

In the parlance of UIG, preference datasets suitable for personalized summarization training and evaluation are of two categories– (i) those which can be directly modeled into a trajectory pool $\mathcal{T}$ (e.g., PENS dataset (Ao et al., 2021)) and (ii) those which lack user trajectories but contain discrete d-nodes, *model-generated* s-nodes (in contrast to user-generated s-nodes as per UIG definition), and *subjective* user feedback in the form of rating and the associated confidence score for that rating (e.g. OpenAI-Reddit dataset (Völske et al., 2017)). We describe the UIG (i.e., the base u-tier) construction method for both types as follows:

PENS-styled Datasets. The construction of UIG is straightforward in the first case and is done in two steps. In the first step, *click* and *skip* interactions in the train dataset are mapped to document nodes (d-nodes) as incoming edges, forming the corresponding u-tier pool $\mathcal{T}$. As an example, for the PENS dataset, the *clkNews* interaction corresponds to a *click* edge and *uclkNews* to a *skip* edge, forming $\mathcal{T}^{\text{PENS}}$. However, PENS dataset lacks user-specific s-nodes (i.e., true interest evolution over time), rendering $\mathcal{T}^{\text{PENS}}$ an *incomplete representation of the user dynamic preference*[2]. We address this issue in the second step, where we *incorporate* surrogate summaries (a chosen sentence from the corresponding d-node) as s-nodes into $\mathcal{T}$ with the addition of *genSumm* and *summGen* edges.

OpenAI-styled Datasets. For the second category of datasets, we first do a pre-construction

[2]It is important to note that despite this, most recent frameworks train on $\mathcal{T}^{\text{PENS}}$ using history or document titles as "pseudo-targets" or via unsupervised learning (Ao et al., 2021; Song et al., 2023; Yang et al., 2023; Lian et al., 2025).

Category	Dimension	Value
Dataset Overview		
General Stats	# Reddit Posts	123,169
	# Subreddits (Domains)	29
	Policy-Generated Summaries	115,579
	Human-Written Summaries	Available
Train + Validation Statistics		
Article Stats	# Reddit Posts	21,111
	# Policies	81
	# Generated Summaries	107,866
	# Annotators	76
	# Summary Pairs Rated	64,832
Validation Subset		
Subset Details	# Reddit Posts	1,038
	# Policies	13
	# Generated Summaries	7,713
	# Annotators	32
Test Dataset (RLHF-Tuned Policies)		
Evaluation	# Evaluated Policies	4
	# Evaluated Reddit Posts	57 (of 1,038)
	Evaluation Method	Indirect benchmarking
Annotation and Feedback		
Feedback	Rating Scale	1–7
	Confidence Scale	1–9
	Avg. Ratings per Annotator	1,176
	Annotation Format	Summary-pair selection

Table 7: **OpenAI TL;DR (Reddit) dataset statistics.** Overview of policy-conditioned summarization data, annotations, and evaluation setup.

classification of clicked and skipped d-nodes for every human rater u_j. This is done based on a simple heuristic of selecting those d-nodes as clicked which has at least one corresponding model-generated summary (note that there can be multiple models) that received a confidence score above a chosen threshold (in the case of OpenAI-Reddit, we chose that to be 6 out of 9). We then select the best model-generated summary (i.e., one with the highest rating given by u_j) as the surrogate expected s-node for u_j. We then randomly sequence all such $(d-s)$-node pairs along with the skipped d-nodes to form τ^{u_j} (thereby $\mathcal{T}^{\text{OAI}}$). This method makes UIG-modeling *compatible with most summarization datasets that are not PENS-styled*.

D UIG Augmentation

After inserting surrogate s-nodes with *genSumm* and *summGen* interactions in the base trajectories, we follow `PerAugy`-style (Chatterjee et al., 2025) augmentation to (i) increase the diversity of user trajectories while preserving realistic temporal structure, and (ii) reduce incoherence in summaries. This augmentation is applied to the u-tier trajectories τ^{u_j} prior to training, yielding a more diverse pool of trajectories used to form $\mathcal{T}^{\text{PENS-D}}$ (and its perturbed variants).

Double Shuffling. Given m sampled training trajectories $\mathcal{T}^{m}_{\text{sample}} \subseteq \mathcal{T}_{\text{train}}$, we select a target τ^{u_j}target and uses the remaining $m-1$ trajectories as sources. This forms a synthetic trajectory $\tau_{\text{S}}^{u_j}$ by (i) preserving an early prefix of the target via an offset O, and (ii) stitching contiguous source segments into later positions. A gap hyperparameter G enforces that between two substitutions, G consecutive target steps remain intact, preserving local temporal realism.

Stochastic Markovian Perturbation. While trajectory shuffling improves diversity, it does not modify incompatible summary nodes (s-nodes) that disrupt temporal coherence. We apply a local context aware summary content perturbation by replacing an s-node with a sentence that better matches the recent context of the *target* user. Consider a substituted summary node $s^{(t_i)}$ in $\tau_{\text{S}}^{u_j}$. Let $\tau_{c_k}^{u_j}$ be a backward context window of the previous k nodes $\{c_q\}_{q=1}^{k}$ (most recent first), and let $d^{(t_{i-1})}$

Characteristic	$\mathcal{T}_{\text{train}}^{\text{PENS-D}}$	$\mathcal{T}_{\text{train}}^{\text{OAI}}$
# u-nodes (trajectories)	60,000	35,000
# d-nodes per trajectory	123.7	36.92
# s-nodes per trajectory	15.10	11.44
Avg. trajectory length	129.8	48.37
Max. trajectory length	200	50
Min. trajectory length	5	25
Rate of Topic Shift	0.77	0.48

Table 8: **User–Interaction Graph (UIG) statistics.** Structural properties of training trajectories derived from augmented PENS and OpenAI TL;DR datasets.

denote the preceding d-node. We split $d^{(t_{i-1})}$ into $n_{d^{(t_{i-1})}}$ candidate sentences $\{st_p\}_{p=1}^{n_{d^{(t_{i-1})}}}$. Using SBERT embeddings, we compute an influence-weighted distance between each candidate sentence and the context window. Temporal relevance is enforced via an exponential decay weight $w_q = \exp(-\lambda \cdot \text{pos}(c_q))$ with $\text{pos}(c_1) = 0$.

The distance matrix is defined $\sigma \in \mathbb{R}^{n_{d^{(t_{i-1})}} \times k}$ as $\sigma_{(p,q)} = \sigma(\mathbf{e}_{st_p}, \mathbf{e}_{c_q})$, where $\sigma(\cdot,\cdot)$ is RMSD. We select:

$$\hat{s}^{(t_i)} = \arg\min \sum_{q=1}^{k} \exp\Big(-\lambda \cdot \text{pos}(c_q)\Big)\, \sigma(\mathbf{e}_{st_p}, \mathbf{e}_{c_q}). \quad (11)$$

Equivalently, the per-sentence scores are given by $\mathbf{sigma\,w}$ where $\mathbf{w}_q = \exp(-\lambda \cdot \text{pos}(c_q))$.

Thus, we obtain $\tau_{\text{P}}^{\hat{u}}$ without leakage of summaries from any testbed. For PENS dataset, $\tau_{\text{SMP}}^{\hat{u}}$ becomes $\tau^{\text{PENS-D}}$ and for OpenAI-Reddit, we get τ^{OAI} for training. The perturbed s-nodes enable richer supervision for personalized summarization tasks.

E Detailed Formulation of the PerDucer Encoder

This appendix provides complete mathematical details of the PerDucer encoder used for Task–1 (Next b-Node Prediction). We expand the b-cell formulation, the Decay-EMA (D-EMA), the contextualization layer (FM-Attn), and the final content-aware MEGA (c-MEGA) state introduced in Section 4.1.

E.1 b-Cell RNN

Each behavioral step $b_{u_j}^{(t_i)} = (hd^{(t_{i-1})}, a^{(t_i)}, tl^{(t_i)})$ is encoded using a three-stage recurrent b-cell. Let the inputs be:

$$\mathbf{e}_{hd}^{(t_i)}, \quad \mathbf{e}_{a}^{(t_i)}, \quad \mathbf{e}_{tl}^{(t_i)}.$$

The b-cell maintains a content state $\mathbf{c}_{tl}^{(t_i)}$ flowing across time.

Head-Cell. The head-cell fuses the previous tail content with the head-node embedding:

$$\mathbf{c}_{hd}^{(t_i)} = \tanh\Big(W_h \mathbf{c}_{tl}^{(t_{i-1})} + \mathbf{b}_h\Big) + \tanh\Big(W_{hd} \mathbf{e}_{hd}^{(t_i)} + \mathbf{b}_{hd}\Big). \quad (12)$$

Action-Cell with Hyperplane Projection. Following the projection idea of TransH (Wang et al., 2014), we embed the action type and project $\mathbf{c}_{hd}^{(t_i)}$ onto the action hyperplane:

$$\mathbf{e'}_{a}^{(t_i)} = \tanh(W_a \mathbf{e}_{a}^{(t_i)} + \mathbf{b}_a), \quad (13)$$

$$\text{proj}_{\mathbf{e'}_{a}^{(t_i)}}(\mathbf{x}) = \mathbf{x} - \frac{\mathbf{e'}_{a}^{(t_i)} \cdot \mathbf{x}}{\|\mathbf{e'}_{a}^{(t_i)}\|_2^2}\, \mathbf{e'}_{a}^{(t_i)}. \quad (14)$$

The action-cell content is:

$$\mathbf{c}_{a}^{(t_i)} = \tanh\left(W_h \cdot \text{proj}_{\mathbf{e'}_{a}^{(t_i)}}\left(\mathbf{c}_{hd}^{(t_i)}\right) + \mathbf{b}_{hd\perp a}\right) + \mathbf{e'}_{a}^{(t_i)}. \quad (15)$$

Tail-Cell. Finally the tail-cell projects the action content onto the tail-node hyperplane:

$$\mathbf{e'}_{tl}^{(t_i)} = \tanh(W_{tl} \mathbf{e}_{tl}^{(t_i)} + \mathbf{b}_{tl}) \quad (16)$$

$$\mathbf{c}_{tl}^{(t_i)} = \tanh\left(W_h \cdot \text{proj}_{\mathbf{e'}_{tl}^{(t_i)}}\left(\mathbf{c}_{a}^{(t_i)}\right) + \mathbf{b}_{a\perp tl}\right) + \mathbf{e'}_{tl}^{(t_i)}. \quad (17)$$

Local b-Node Embedding. The b-cell emits:

$$\mathbf{e}_{b_{u_j}}^{(t_i)} = \tanh\Big(W_b \mathbf{c}_{tl}^{(t_i)} + \mathbf{b}_b\Big). \quad (18)$$

This yields the local behavioral semantics used in Sec. 4.1.

E.2 Decay-based Exponential Moving Average (D-EMA)

To capture slow preference drift across a user's timeline, PerDucer maintains a cumulative snapshot $\mathbf{z}_{b_{u_j}}^{(1:t_i)}$ updated as:

$$\mathbf{z}_{b_{u_j}}^{(1:t_i)} = \alpha^{(t_i)} \odot \mathbf{e}_{b_{u_j}}^{(t_i)} + \Big(1 - \alpha^{(t_i)} \odot \delta^{(t_i)}\Big) \odot \mathbf{z}_{b_{u_j}}^{(1:t_{i-1})}. \quad (19)$$

Algorithm 1 UIG Construction

function CONSTRUCT_UIG(train_data, test_data, dataset_type)
  Initialize $\mathcal{T}_{\text{PENS}} \leftarrow \emptyset$, $\mathcal{T}_{\text{OAI}} \leftarrow \emptyset$
  for each user u in train_data **do**
    Initialize $\tau_P^u \leftarrow \emptyset$, $\tau_{OAI}^u \leftarrow \emptyset$
    for each interaction in user u's data **do**
      if dataset_type is PENS **then**
        if interaction is *clkNews* **then**
          Map to d-node with a *click* edge
        else if interaction is *uclkNews* **then**
          Map to d-node with a *skip* edge
        end if
        Append mapped d-node to τ_P^u
      else
        if model-generated summary rating < 6 **then**
          Map to d-node with a *skip* edge
        else if model-generated summary rating > 6 **then**
          Map to d-node with a *click* edge
        end if
        if confidence for rating $=$ max **then**
          Map to d-node with a *gensum* edge
          Map to s-node with a *sumgen* edge
        end if
        Append mapped d-node to τ_{OAI}^u
      end if
    end for
    if dataset_type is PENS **then**
      Add τ_P^u to $\mathcal{T}_{\text{PENS}}$
    else
      Add τ_{OAI}^u to $\mathcal{T}_{\text{OAI}}$
    end if
  end for
  if dataset_type is PENS **then**
    for each trajectory τ_P^u in $\mathcal{T}_{\text{PENS}}$ **do**
      Retrieve corresponding s-nodes from test_data at associated time-steps
      Insert s-nodes into τ_P^u using *genSumm* and *sumgen* edges
    end for
    $\mathcal{T}^{\text{PENS-D}} \leftarrow \mathcal{T}_{\text{PENS}}$
    return $\mathcal{T}^{\text{PENS-D}}$
  else
    return $\mathcal{T}_{\text{OAI}}$
  end if
end function=0

Algorithm 2 Double Shuffling

Require: Target trajectory $\tau_{\text{target}}^{u_j}$; source trajectories $\{\tau_{\text{source}}^{u_i}\}_{i=1}^{m-1}$; offset distribution $\mathcal{P}_O$; gap G
Ensure: Synthetic trajectory $\tau_{\text{S}}^{u_j}$
Sample $O \sim \mathcal{P}_O$; $\tau_{\text{sS}}^{u_j} \leftarrow \tau_{\text{target}}^{u_j}[1{:}O]$
$t \leftarrow O$
for $i = 1$ to $m-1$ **do**
  Sample a contiguous segment $\tau_{\text{seg}}^{u_i}$ from $\tau_{\text{source}}^{u_i}$ with length l_{seg_i}
  Substitute: replace $\tau_{\text{target}}^{u_j}[t{+}1{:}t{+}l_{\text{seg}_i}]$ with $\tau_{\text{seg}}^{u_i}$
  Append G intact steps from target (if available): $\tau_{\text{target}}^{u_j}[t{+}l_{\text{seg}_i}{+}1{:}t{+}l_{\text{seg}_i}{+}G]$
  $t \leftarrow t + l_{\text{seg}_i} + G$
end for
Append the remaining suffix of $\tau_{\text{target}}^{u_j}$ (if any) and return $\tau_{\text{DS}}^{u_j}$ =0

Algorithm 3 Summary Perturbation

Require: Trajectory $\tau_{\text{DS}}^{u_j}$; window size k; decay λ; perturb prob. p_{SMP}
Ensure: Perturbed trajectory $\tau_{\text{P}}^{u_j}$
$\tau_{\text{P}}^{u_j} \leftarrow \tau_{\text{DS}}^{u_j}$
for each substituted s-node $s^{(t_i)}$ in $\tau_{\text{P}}^{u_j}$ **do**
  Sample $z \sim \text{Bernoulli}(p_{\text{P}})$
  if $z = 1$ **then**
    Build context window $\{c_q\}_{q=1}^{k}$ and candidate sentences $\{st_p\}$
    Compute $\hat{s}^{(t_i)}$ via Eq. equation 11
    Replace $s^{(t_i)} \leftarrow \hat{s}^{(t_i)}$
  end if
end for
return $\tau_{\text{P}}^{u_j}$ =0

The two gates combine as $\tilde{\alpha}^{(t_i)} = \alpha^{(t_i)} \odot \delta^{(t_i)}$, and are content-aware, which are computed as:

Decay Gate.

$$\alpha^{(t_i)} = \tanh\left(W_\alpha \left[\mathbf{z}_{b_{u_j}}^{(1:t_{i-1})}; \mathbf{e}_{b_{u_j}}^{(t_i)}\right] + \mathbf{b}_\alpha\right). \tag{20}$$

Damping Gate.

$$\delta^{(t_i)} = \tanh\left(W_\delta \left[\mathbf{z}_{b_{u_j}}^{(1:t_{i-1})}; \mathbf{e}_{b_{u_j}}^{(t_i)}\right] + \mathbf{b}_\delta\right). \tag{21}$$

This full version corresponds to the abbreviated formula in Sec. 4.1.

E.3 Forward-Masked Self-Attention (FM-Attn)

To capture non-local recurrences and thematic cycles, we compute contextualized snapshots using forward-masked attention. Let:

$$\mathbf{Z}_{1:i} = [\mathbf{z}_{b_{u_j}}^{(1:t_1)}, \ldots, \mathbf{z}_{b_{u_j}}^{(1:t_i)}].$$

We first obtain queries, keys, and values:

$$\mathbf{Q}_{1:i} = \mathbf{Z}_{1:i} W_Q, \tag{22}$$
$$\mathbf{K}_{1:i} = \mathbf{Z}_{1:i} W_K, \tag{23}$$
$$\mathbf{V}_{1:i} = \mathbf{Z}_{1:i} W_V. \tag{24}$$

The forward mask ensures that position i attends only to $1:i$:

$$\mathbf{A}_{i,j} = \begin{cases} \frac{1}{\sqrt{d}} \left(\mathbf{Q}_i \cdot \mathbf{K}_j^\top \right), & j \leq i, \\ -\infty, & j > i. \end{cases} \tag{25}$$

The contextualized snapshot is:

$$\mathbf{c}_{b_{u_j}}^{(t_i)} = \mathrm{softmax}(\mathbf{A}_{i,*})\, \mathbf{V}_{1:i}. \tag{26}$$

This corresponds to the shorthand -$(\cdot)$ in Sec. 4.1.

E.4 Content-Aware MEGA Fusion (c-MEGA)

The contextualized state from FM-Attn is fused with the current local embedding to produce the c-MEGA representation.

Input Gate.

$$\gamma^{(t_i)} = \sigma\left(W_i \mathbf{z}_{b_{u_j}}^{(1:t_i)} + \mathbf{b}_i \right). \tag{27}$$

Final Fusion.

$$\mathbf{h}_{b_{u_j}}^{(t_i)} = \gamma^{(t_i)} \odot \mathbf{c}_{b_{u_j}}^{(t_i)} + \left(1 - \gamma^{(t_i)}\right) \odot \mathbf{e}_{b_{u_j}}^{(t_i)}. \tag{28}$$

This recovers the full expression summarized in Sec. 4.1.

E.5 Prediction Head

The final contextualized embedding at the last step t_l is used to predict the next b-node:

$$\mathbf{h}_{b_{(\mathrm{q},u_j)}}^{(t_{l+1})} = W_{\mathrm{pred}}\, \mathbf{h}_{b_{u_j}}^{(t_l)} + \mathbf{b}_{\mathrm{pred}}. \tag{29}$$

F Baselines

F.1 Baseline LLMs

1. Zephyr 7B β. Zephyr(Tunstall et al., 2023) is a 7B-parameter transformer model fine-tuned from Mistral-7B using Direct Preference Optimization (DPO) on publicly available and synthetic data. It removes some traditional alignment constraints to improve raw performance, achieving strong results on benchmarks like MT-Bench (7.34 vs. 6.86 for LLaMA2-70B-Chat). Zephyr is optimized for helpful dialogue and is openly available under an MIT license. Its design focuses on efficiency and high-quality responses without relying on reinforcement learning from human feedback.

Mistral 7B. Mistral-Instruct(Jiang et al., 2023) is a dense transformer model using grouped-query attention (GQA) and sliding window attention (SWA) to efficiently scale with long context inputs. Pre-trained on around 2 trillion tokens, it delivers strong performance across NLP and coding benchmarks and surpasses larger models like LLaMA2-13B in many areas. It is fully open-source (Apache 2.0) and includes an instruction-tuned variant, making it widely adopted for fine-tuning and deployment.

LLaMA 2 13B. LLaMA-2(Touvron et al., 2023) LLaMA 2 13B by Meta is a 13B-parameter autoregressive transformer trained on 2 trillion tokens of public data, with a context length of 4096. It supports chat via instruction tuning and RLHF. Though once state-of-the-art among open models, newer models like Mistral 7B now outperform it in many tasks. LLaMA 2 remains a strong, widely used foundation model with full documentation and open access under Meta's license.

DeepSeek-R1 14B. DeepSeek-R1(DeepSeek-AI et al., 2025) is a 14.8B-parameter model distilled from Qwen 2.5-14B, specifically optimized for math, code, and reasoning tasks. It was fine-tuned on 800K examples generated by a larger DeepSeek R1 model and is released under an MIT license. Despite being smaller, it rivals much larger models on benchmarks like AIME and MATH, offering strong step-by-step reasoning while remaining efficient and open for further customization.

F.2 Baseline SLMs

SmolLM2-1.7B. SmolLM2 (Allal et al., 2025) is a lightweight language model with 1.7B parameters, designed for efficient performance on de-

vices with limited resources. It offers fast inference and handles common NLP tasks well, making it a strong baseline for compact models. SmolLM2 was trained primarily on a mix of general-domain text tasks, including language modeling, next-word prediction, and basic text classification. The training involved supervised learning on curated datasets combined with unsupervised pretraining on large text corpora to build foundational language understanding while keeping the model compact.

Qwen2.5-0.5B Qwen2.5 (Qwen et al., 2025) is a smaller language model of 0.5B parameters, that balances scale and performance. It delivers better accuracy and versatility across NLP tasks, serving as a solid baseline for research and development without requiring massive computing power. Qwen2.5 was trained on a broader and more diverse set of tasks such as language modeling, question answering, summarization, and dialogue generation. It used a combination of large-scale unsupervised pretraining on extensive text data followed by supervised fine-tuning on specific downstream tasks to improve accuracy and contextual comprehension.

F.3 Baseline Generic Summarizers

1. BigBirdPegasus. BigbirdPegasus, proposed by (Zaheer et al., 2020) is an extension of Transformer based models designed specifically for processing longer sequences. It utilizes sparse attention, global attention, and random attention mechanisms to approximate full attention. This enables BigBird to handle longer contexts more efficiently and, therefore, can be suitable for summarization.

2. SimCLS. A Simple Framework for Contrastive Learning of Abstractive Summarization (Liu and Liu, 2021) uses a two-stage training procedure. In the first stage, a Seq2Seq model (Lewis et al., 2020) is trained to generate candidate summaries with MLE loss. Next, the evaluation model, initiated with RoBERTa is trained to rank the generated candidates with contrastive learning.

F.4 Baseline Personalized Models

PENS-NRMS Injection-Type 1. The PENS framework (Ao et al., 2021) generates personalized summaries by incorporating user embeddings along with the input news article. For this variant, user embeddings are derived using NRMS (Neural News Recommendation with Multi-Head Self-Attention) (Wu et al., 2019b), which includes a multi-head self-attention based news encoder to represent news titles, and a user encoder that captures browsing behavior through multi-head self-attention over clicked articles. Additive attention mechanisms are employed to highlight important words and articles. In Injection-Type 1, the NRMS user embedding is injected by initializing the decoder's hidden state, thereby directly influencing the summary generation process from the start.

PENS-NRMS Injection-Type 2. This variant also uses NRMS for user embedding, but personalization is introduced differently. Instead of initializing the decoder, the user embedding is injected into the attention mechanism of the PENS model. This modulates the attention weights over the news body, enabling the model to focus on content aligned with the user's preferences.

PENS-NAML Injection-Type 1. NAML (Neural News Recommendation with Attentive Multi-View Learning) (Wu et al., 2019a) generates news representations by attending over multiple views, including titles, bodies, and topic categories. The user encoder learns from interacted news and selects the most informative content for personalization. The resulting user embedding is integrated into the PENS decoder using Injection-Type 1, i.e., by initializing the decoder's hidden state.

PENS-EBNR Injection-Type 1. EBNR (Embedding-based News Recommendation) (Okura et al., 2017) models user preferences using an RNN over browsing histories to produce user embeddings. These embeddings are injected into the PENS model via Injection-Type 1 by initializing the decoder, thereby influencing the initial decoding steps with user-specific information.

PENS-EBNR Injection-Type 2. This configuration uses the same user encoder from EBNR but applies Injection-Type 2. Here, the user embedding is incorporated into the decoder's attention layers, allowing the model to personalize attention distributions over the news body during decoding.

General Then Personal (GTP). General Then Personal (GTP) (Song et al., 2023) is a two-stage framework for personalized headline generation. In stage-1, a Transformer-based encoder–decoder model is pre-trained on large-scale news article–headline pairs to learn robust, content-focused headline generation without personalization. In stage-2, a separate "headline customizer" refines

the general headline by incorporating user-specific preferences, which are encoded as a control code by the user encoder TrRMIo. To bridge the gap between general generation and personalized refinement, GTP introduces two mechanisms: (i) **Information Self-Boosting (ISB)**, which reintroduces relevant content details from the article to prevent information loss during customization; and (ii) **Masked User Modeling (MUM)**, which randomly masks parts of the user embedding during training and reconstructs them, reducing the model's over-reliance on its general parameters.

Signature Phrase. Another line of personalization focuses on condensing a user's reading history into a collection of *signature phrases* (Cai et al., 2023). These phrases, derived through contrastive learning over news articles without annotated data, act as dynamic user profiles that adapt as interests evolve. Such phrases need not appear verbatim in the user's history but instead encode higher-level signals. Using these phrases, the model learns to generate personalized headlines that connect new articles with the user's inferred interests, yielding outputs that are engaging, relevant, and grounded in article content rather than drifting toward clickbait.

G Training Details

G.1 Compute Resources

All preprocessing and embedding tasks were run on CPU-only machines, while model training utilized cloud-based GPUs. We utilized 16GB CPU cores for seeding embeddings with PromptRank-T5 on each node, for extracting keyphrase vocabulary with YAKE across all d-nodes, and for generating keyphrase ground-truth (distribution of keyphrases) for s-nodes using spaCy3.7. The training of each version of `PerDucer`, inferencing, and computing results were run with mixed-precision (FP16) training on NVIDIA L40 and L40S GPUs[3], alongside CPU-based preprocessing and data loading.

G.2 Training

`PerDucer` was trained end-to-end for 6 epochs. A batch size of 128 was used throughout, and optimization employed PyTorch's AdamW[4] with learning rate 3×10^{-4}, betas (0.9, 0.999), epsilon $\times10^{-8}$, weight decay 0.01, a fixed learning rate policy, and dropout probability 0.1 on all self-attention and feed-forward layers. Total training steps were computed as $(N_{\text{train}}/128) \times 6$, where N_{train} is the size of the training set. The vocabulary of keyphrases from training data is approximately 176K, and the average number of keyphrases extracted from each s-node is 20. The total number of behaviors in the training data is 547K.

[3]We gratefully acknowledge Lightning.ai for providing virtual compute resources using L40 and L40S GPUs.

[4]AdamW implementation: `torch.optim.AdamW` (version 1.13.1)

H Detailed Results

H.1 Personalization Boosting in LLMs (MS/CAS PENS)

On MS/CAS PENS, the **B-tier Vanilla** models achieve near parity with the strongest prompt-only baselines, despite relying on a structured RNN-style behavioral encoder. Across LLMs, B-tier Vanilla yields PSE-JSD values in the range of **0.202–0.211**, compared to **0.217–0.238** for the **2-shot History** setting. Similarly, B-tier Vanilla achieves PSE-SU4 scores of **0.088–0.093** and PSE-METEOR scores of **0.098–0.105**, closely matching or exceeding the corresponding 2-shot History ranges of **0.071–0.087** (SU4) and **0.078–0.102** (METEOR). This indicates that lightweight behavioral encoding can substitute for carefully engineered prompt-based history injection, while remaining model-agnostic and more stable.

Introducing explicit temporal behavior modeling via **D-EMA** leads to a clear and consistent improvement over B-tier Vanilla across all LLM backbones on PENS. D-EMA reduces PSE-JSD further to **0.296–0.321** and improves PSE-SU4 to **0.103–0.114**, with corresponding PSE-METEOR scores in the range of **0.284–0.311**. This confirms that temporally aligned memory is critical for capturing evolving user preferences beyond static interaction summaries.

Augmenting D-EMA with **FM-Attn (c-EMA)** produces additional but modest gains. Across models, D-EMA + FM-Attn achieves PSE-JSD values of **0.317–0.329**, SU4 values of **0.114–0.124**, and METEOR scores of **0.307–0.316**, consistently matching or slightly improving upon D-EMA. These results suggest that controlled feature mixing refines personalization signals without destabilizing temporal representations.

Finally, the full **c-MEGA** model delivers the strongest personalization performance on MS/CAS PENS. c-MEGA achieves the best PSE-JSD scores across all LLMs (**0.327–0.348**), along with the highest SU4 (**0.124–0.134**) and METEOR (**0.315–**

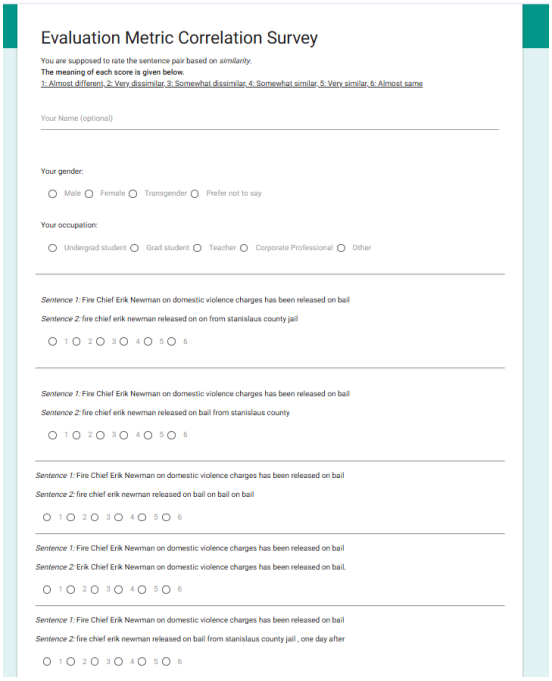

Figure 5: Survey Template form for human-based qualitative assessment of model-generated summaries. Annotators were mostly students(50), and Faculties (4).

0.324) values among all evaluated settings. This establishes c-MEGA as the most effective configuration, benefiting jointly from temporal memory, feature-level aggregation, and cross-context behavioral fusion.

Detailed per-model MS/CAS PENS results are reported in Table 12.

Ablation: Influence of LLM Temperature. We ablate and observe the effect of varying the LLM decoding temperature ($T \in \{0.2, 0.5, 0.8\}$) on personalization performance. Across all evaluated models, increasing temperature leads to a *consistent trade-off* across PSE metrics (Table 13). Specifically, higher temperatures systematically reduce PSE-JSD (e.g., from **0.342–0.348** at T=0.2 to **0.236–0.266** at T=0.8), indicating greater lexical diversity relative to the reference. This also results in degraded semantic alignment, with PSE-SU4 dropping from **0.128–0.134** to **0.075–0.080**, and PSE-METEOR decreasing from **0.316–0.324** to **0.187–0.213** as temperature increases. Overall, higher temperatures dilute keyphrase-conditioned semantic fidelity, confirming that lower-temperature decoding is preferable for stable personalized summarization.

Ablation: Influence of Number of Keyphrases. We analyze the effect of varying the number of injected keyphrases ($k \in \{5, 10, 15\}$) on personalization performance. Using a small number of keyphrases (k=5) results in consistently weak personalization signals across all LLMs, with PSE-JSD confined to **0.063–0.077**, PSE-SU4 to **0.037–0.048**, and PSE-METEOR to **0.039–0.055** (Table 3). This indicates insufficient behavioral grounding when too few keyphrases are provided. Increasing the keyphrase budget to k=10 yields a substantial improvement across all metrics and models. At this setting, PSE-JSD rises sharply to **0.327–0.348**, accompanied by corresponding gains in PSE-SU4 (**0.124–0.134**) and PSE-METEOR (**0.315–0.324**). These results suggest that k=10 keyphrases provide an effective balance between personalization signal strength and semantic coherence. Further increasing the number of keyphrases to k=15 leads to a consistent but modest degradation in performance. Across LLMs, PSE-JSD decreases to **0.300–0.318**, PSE-SU4 to **0.114–0.123**, and PSE-METEOR to **0.287–0.295**. This drop suggests diminishing returns and mild noise injection when the prompt becomes overly saturated with keyphrases. Overall, the results indicate a clear non-monotonic trend, with k=10 emerging as the optimal operating point for keyphrase-guided personalization in `PerDucer`.

Ablation: Seed Embedding via SBERT. We study the effect of seed embedding quality by initializing c-MEGA with SBERT embeddings (Reimers and Gurevych, 2019) instead of learned PromptRank-based representations. Across all evaluated LLMs and both datasets, **SBERT-seeded c-MEGA consistently underperforms the fully learned c-MEGA variant** on all PSE metrics. On MS/CAS PENS, SBERT seeding results in lower PSE-JSD scores in the range of **0.214–0.244**, compared to **0.327–0.348** achieved by c-MEGA. Correspondingly, PSE-SU4 drops from **0.124–0.134** (c-MEGA) to **0.079–0.093**, and PSE-METEOR decreases from **0.315–0.324** to **0.221–0.241**. Similar degradation trends are observed on the OpenAI Reddit test set across all backbone models. These results confirm that while SBERT provides a reasonable semantic initialization, it is suboptimal for `PerDucer`, whose downstream objective is *keyphrase extraction and behavioral alignment*. PromptRank-based seed embeddings, being explicitly trained for ranking and phrase salience, produce representations that are substantially better aligned with the personalization objective, lead-

ing to consistently stronger performance. Detailed per-model comparisons are reported in Table 12.

I Human Judgement

Survey-based Direct Human Evaluation. We conducted a human-based qualitative evaluation to assess the quality of model-generated summaries with respect to gold reference summaries. The evaluation involved 54 annotators, comprising 50 graduate students and 4 faculty members, all proficient in English and experienced in reading technical or news-style content. For each test instance, annotators were shown the source document, the gold reference summary, and seven predicted summaries generated by different models. To mitigate bias, model identities were anonymized and the order of predicted summaries was randomized for each instance. Annotators were asked to independently rate each predicted summary relative to the gold reference along multiple dimensions, including relevance (coverage of salient information), coherence (logical flow and readability), faithfulness (absence of hallucinations or contradictions with the source), and overall quality. Ratings were collected using a fixed Likert scale, and no time constraints were imposed during the annotation process.

Human-Judgment Interpolation from OpenAI-Reddit dataset. The interpolation of human judgment scores is performed by leveraging the OpenAI-Reddit dataset, which provides multiple human-rated summaries for each article. For every article, the highest-rated human summaries which are 7 are designated as the *benchmark reference*. All candidate summaries, including the benchmark, are first embedded into a high-dimensional semantic space using a SentenceTransformer (Reimers and Gurevych, 2019) model. The semantic deviation between the benchmark embedding V_b and any other summary embedding V_o is quantified via the Root Mean Square Deviation (RMSD), which in this context is equivalent to the Euclidean distance:

$$\text{RMSD}(V_b, V_o) = \sqrt{\sum_{i=1}^{n} (b_i - o_i)^2}\,.$$

In practice, this computation is implemented efficiently using NumPy's linear algebra module, `np.linalg.norm`. The resulting RMSD values are then grouped according to the original human rating of each summary (e.g., $7/7$, $6/7$). By averaging the RMSD values within each rating group, we obtain a mapping between human-judged quality scores and embedding-space distances. Notably, the RMSD for summaries rated $7/7$ is not always zero, as there may exist multiple distinct summaries with a top score for the same article; while all such summaries are judged as equally high-quality by humans, their semantic embeddings can still differ due to variations in phrasing, emphasis, or lexical choices. These aggregated averages form the scoring thresholds used for interpolating human judgment in our evaluation framework.

J Prompt Template

As discussed in Section 4.3, we contrast our `PerDucer`-guided summarization with 0/2-shot user-history prompting and prompt-chaining approaches that condition LLMs directly on past interactions. In our setting, we provide a structured user context by leveraging $\mathcal{T}_{\text{test}}^{\text{PENS-D}}$ and $\mathcal{T}_{\text{test}}^{\text{OAI}}$ as user histories for preference induction. In contrast, for `PerDucer`-guided generation, we supply the LLM only with the main article and the extracted keyphrases predicted by `PerDucer`, which act as explicit preference cues for controlled summarization. To ensure consistent and faithful comparisons across prompting strategies, all LLM-based generations use a fixed decoding configuration, with temperature set to 0.2, nucleus sampling with top-p set to 0.9, and a maximum output length of 25 tokens. This configuration prioritizes faithfulness and reduces variability introduced by stochastic decoding, allowing differences in output quality to be attributed primarily to the conditioning strategy rather than decoding randomness. The detailed prompt structures for all compared methods are illustrated in Figure 6.

K License and Usage Statement

In this work, we utilize the following pre-trained large language models (PLMs) and small language models (SLMs):

- LLMs: DeepSeek-R1 14B (MIT License), Mistral-7B-Instruct (Apache 2.0), LLaMA2-13B (Llama 2 Community License), and Zephyr 7B (β) (MIT License).
- SLMs: SmolLM2 1.7B (Apache 2.0) and Qwen2.5 0.5B (Apache 2.0).

All models are used according to their respective licenses and terms provided by their original cre-

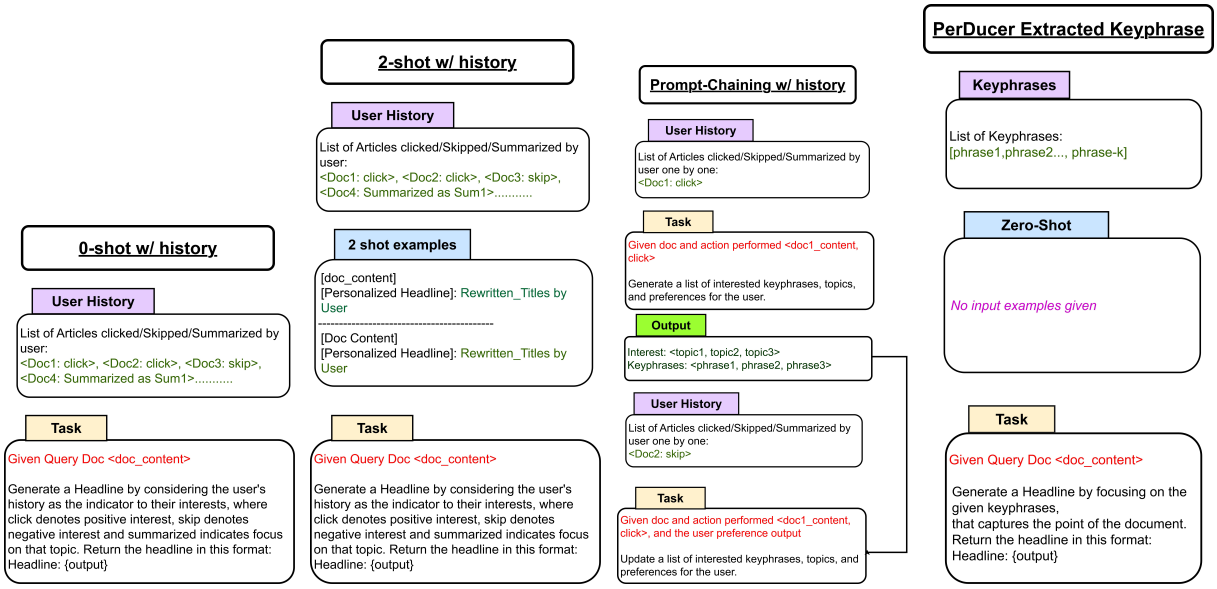

Figure 6: **Comparison of prompting strategies for personalized summarization.** From left to right: **(a)** 0-shot prompting, **(b)** 2-shot prompting (Patel et al., 2024), **(c)** chain-based prompting with intermediate reasoning cues, and **(d)** PerDucer top-k key-phrase guidance, where predicted personalized key phrases are injected as explicit control signals into frozen LLM/SLM summarizers.

ators. Proper attribution is given to each model's developers as cited in our references.

We also use the following datasets:

- **MS/CAS PENS dataset:** We comply with the dataset's terms of use, which is derived from the Microsoft Research License (https://github.com/msnews/MIND/blob/master/MSR%20License_Data.pdf).
- **OpenAI Reddit dataset:** We comply with the MIT License specifications as set by OpenAI (https://github.com/openai/summarize-from-feedback/blob/master/LICENSE)

We have ensured that all datasets and models are used responsibly, respecting privacy, consent, and ethical guidelines. When applicable, data is anonymized and handled according to the ethical standards of ACL.

Symbol	Description
User–Interaction Graph (u-tier) and trajectories	
UIG $G = \langle N, E \rangle$	User–Interaction Graph modeled as a DAG with nodes N and edges E
$u_j^{(t_0)}$	j-th user node (u-node) at initial time t_0
$d^{(t_p)}$	Document/news node (d-node) interacted at time-step t_p
$s_j^{(t_q)}$	User-specific expected summary/headline node (s-node) at time-step t_q (gold or model-generated)
$a_d^{(t_p)}$	Document-level interaction edge into a d-node at t_p (e.g., *click/skip/genSumm*)
$a_s^{(t_q)}$	Follow-up edge from a d-node to its s-node at t_q (e.g., *summGen*)
τ^{u_j}	u-tier interaction trajectory (time-ordered path) of user u_j
$\mathcal{T}$	Set (pool) of all user trajectories in the UIG
$\mathcal{T}_{\text{train}}, \mathcal{T}_{\text{test}}$	Training/test splits of the trajectory pool
$\mathcal{T}^{\text{PENS}}$	Trajectory pool derived from the PENS dataset
$\mathcal{T}^{\text{PENS-D}}$	PENS-derived trajectories augmented with inserted test s-nodes
$\mathcal{T}^{\text{OAI}}$	UIG-modeled trajectory pool from OpenAI–Reddit-style datasets
Behavior triples and b-tier abstraction	
$b_{u_j}^{(t_i)} = \langle hd^{(t_{i-1})},\ a^{(t_i)},\ tl^{(t_i)} \rangle$	Behavior triple at time t_i (head node, action, tail node)
$hd^{(t_{i-1})}$	Head node of the behavior triple (previous d-/s-node)
$tl^{(t_i)}$	Tail node of the behavior triple (resulting d-/s-node)
$a^{(t_i)}$	Action type connecting head and tail at step t_i
$\tau_b^{u_j}$	b-tier trajectory of user u_j (sequence of behavior triples / b-nodes)
$b_{(q,u_j)}$	Query behavior triple (next-step target) for user u_j and query document d_q
Seeding and local b-node composition (b-cell)	
$\mathbf{e}_{u_j}^{(t_0)}$	Initial user embedding (title embedding of the first document; cold-start mitigation)
$\mathbf{e}_{hd}^{(t_{i-1})}$	Embedding of the head node at time t_{i-1} (PromptRank-KPE seeded)
$\mathbf{e}_{tl}^{(t_i)}$	Embedding of the tail node at time t_i (PromptRank-KPE seeded)
$\mathbf{e}_a^{(t_i)}$	Action-edge embedding at time t_i (seeded as 4-d one-hot)
$g_b(\cdot)$	TransH-inspired learned composition mapping $(\mathbf{e}_{hd}, \mathbf{e}_a, \mathbf{e}_{tl}) \mapsto \mathbf{e}_{b_{u_j}}^{(t_i)}$
$\mathbf{c}_{hd}^{(t_i)}$	Head-cell content at time t_i
$\mathbf{c}_a^{(t_i)}$	Action-cell content at time t_i
$\mathbf{c}_{tl}^{(t_i)}$	Tail-cell content at time t_i (recurrent content flowing across time)
$\text{proj}_{\mathbf{e}_x'^{(t_i)}}(\cdot)$	Hyperplane projection operator used in b-cell ($x \in \{a, tl\}$)
$\mathbf{e}_a'^{(t_i)}, \mathbf{e}_{tl}'^{(t_i)}$	Projected action embedding and projected tail embedding
$W_h, W_{hd}, W_a, W_{tl}, W_b$	Learnable weight matrices in the b-cell (Appendix C)
$\mathbf{b}_h, \mathbf{b}_{hd}, \mathbf{b}_a, \mathbf{b}_{tl}, \mathbf{b}_b$	Bias vectors associated with b-cell projections
History-aware encoding (D-EMA, FM-Attn, c-MEGA)	
$\mathbf{e}_{b_{u_j}}^{(t_i)}$	Local b-node embedding at step t_i (b-cell output)
$\mathbf{z}_{b_{u_j}}^{(1:t_i)}$	D-EMA cumulative snapshot up to step t_i
$\boldsymbol{\alpha}^{(t_i)}$	Content-aware decay gate for updating D-EMA
$\boldsymbol{\delta}^{(t_i)}$	Content-aware damping gate (controls carryover of past snapshot)
$\tilde{\boldsymbol{\alpha}}^{(t_i)} = \boldsymbol{\alpha}^{(t_i)} \odot \boldsymbol{\delta}^{(t_i)}$	Effective past gate used in the D-EMA recursion
$\text{FM-Attn}(\cdot)$	Forward-masked self-attention over $\{\mathbf{z}^{(1:t)}\}$ (non-local/cyclical dependencies)
$\mathbf{c}_{b_{u_j}}^{(t_i)}$	Contextualized snapshot from FM-Attn
$\boldsymbol{\gamma}^{(t_i)}$	Input gate for fusing contextual signal and local embedding
$\mathbf{h}_{b_{u_j}}^{(t_i)}$	c-MEGA state (gated fusion of $\mathbf{c}_{b_{u_j}}^{(t_i)}$ and $\mathbf{e}_{b_{u_j}}^{(t_i)}$)
$\mathbf{h}_{b_q u_j}^{(t_{l+1})}$	Predicted next (query) b-node embedding from the prediction head/MLP
Keyphrase decoder and training objective	
$\hat{\mathbf{P}}_{KP}$	Predicted keyphrase relevance distribution over the global keyphrase vocabulary
$W_{KP},\ \phi(\cdot)$	Keyphrase scoring head and projection into a keyphrase latent space
$\{kp_i\}_{1:k},\ k$	Top-k predicted keyphrases used as control cues for guided summarization
$\mathcal{L}_{\text{KPE}}$	Keyphrase prediction loss (BCE over multi-hot keyphrase labels)
$\hat{\mathbf{P}}_{\text{pos}},\ W_{\text{pos}}$	Position distribution and position head for temporal alignment
$\mathcal{L}_{\text{ENC}}$	Temporal alignment loss (NLL over step index)
$\mathcal{L}_{\texttt{PerDucer}}$	Final training objective (paper uses $0.5\,\mathcal{L}_{\text{KPE}} + 0.5\,\mathcal{L}_{\text{ENC}}$)
Dataset-specific edge labels	
clkNews, *uclkNews*	Clicked and unclicked news items in PENS logs
genSumm, *summGen*	Summary-generation edge types used to insert/link s-nodes in augmented UIGs

Table 9: **Notation summary.** Symbols used for UIG construction, behavior trajectories, PerDucer encoder states (D-EMA/FM-Attn/c-MEGA), and the keyphrase decoder/training objectives.

Metric	PerDucer (Ours)	DeepSeek-R1 (2-shot)	Relative Gain
Trainable parameters	520M	14B	32.6× fewer
Deployable parameters	590M	14B	24.6× fewer
Avg. output length	8–12 keyphrases	20 tokens	–
FLOPs / output (est.)	6.1×10^{10}	1.6×10^{12}	∼26× lower
Inference time / sample (est.)	0.4–3 s	15–160 s	10–50× faster
GPU cost (training)	22 GPU-hours	42 GPU-hours	∼2× lower
VRAM footprint (deploy)	∼1.2 GB	>28 GB	edge-deployable
LLM finetuning required	**No**	**Yes, for better results**	avoids LLM retraining
Prompt tokens injected	20–40	130	minimal overhead

Table 10: Training and deployment resource comparison between PerDucer and the strongest LLM baseline (DeepSeek-R1). PerDucer separates lightweight keyphrase generation from frozen LLM inference.

Component / Parameter	Shape / Value
Embedding and state dimensions	
Seed embedding dimension	$d = 768$ (PromptRank KPE)
Hidden / memory dimension	768 (shared across encoder)
Action encoding	4-d one-hot vector
b-cell projection matrices	
$W_h,\ W_{hd},\ W_{tl},\ W_b$	768×768
W_a	768×4
D-EMA gating (paper: D-EMA)	
$W_\alpha,\ W_\delta$	1536×768
c-MEGA contextual attention	
$W_Q,\ W_K,\ W_V,\ W_i$	768×768
Prediction and alignment heads	
Behavior prediction head W_{pred}	$\lvert\mathcal{V}_{\text{B}}\rvert \times 768$
Position alignment MLP W_{pos}	$768 \to 768 \to 768$
Global keyphrase decoder (Task-3)	
Fusion projection W_f	768×512
Keyphrase scorer W_{KP}	$\lvert\mathcal{V}_{\text{KP}}\rvert \times 512$
Loss	Multi-label BCE over $\mathcal{V}_{\text{KP}}$
Note: $\mathcal{V}_{\text{KP}}$ (global keyphrase vocabulary) and $\mathcal{V}_{\text{B}}$ (behavior vocabulary) are explicitly defined in the paper. All listed shapes follow directly from the paper's stated 768-d seeding and shared hidden state assumption. No additional dimensions are introduced.	

Table 11: **PerDucer: architecture parameters and learned weight dimensions.**

Context Source	Model	MS/CAS PENS Test			OpenAI Reddit Test		
		PSE-JSD	PSE-SU4	PSE-MET	PSE-JSD	PSE-SU4	PSE-MET
History Prompt-Chaining	Mistral-7B	0.072	0.026	0.023	0.045	0.014	0.017
	DeepSeek-R1	0.078	0.028	0.027	0.051	0.019	0.022
	Zephyr-7B-β	NA*	NA*	NA*	NA*	NA*	NA*
	LLaMA-13B	NA*	NA*	NA*	NA*	NA*	NA*
	Qwen2.5-0.5B	NA*	NA*	NA*	NA*	NA*	NA*
	smolLM2-1.7B	NA*	NA*	NA*	NA*	NA*	NA*
2-shot History	Mistral-7B	0.226	0.086	0.083	0.226	0.088	0.103
	DeepSeek-R1	0.238	0.087	0.102	0.243	0.095	0.109
	Zephyr-7B-β	0.231	0.078	0.079	0.214	0.087	0.104
	LLaMA-13B	0.217	0.071	0.078	0.232	0.093	0.107
	Qwen2.5-0.5B	NA*	NA*	NA*	NA*	NA*	NA*
	smolLM2-1.7B	NA*	NA*	NA*	NA*	NA*	NA*
B-tier RNN	Mistral-7B	0.211	0.089	0.097	0.174	0.082	0.089
	DeepSeek-R1	0.215	0.092	0.108	0.177	**0.089**	0.101
	Zephyr-7B-β	0.205	0.088	0.103	0.171	0.081	0.096
	LLaMA-13B	0.206	0.093	0.105	0.161	0.079	**0.093**
	Qwen2.5-0.5B	0.202	0.089	0.101	0.168	0.081	0.095
	smolLM2-1.7B	0.211	0.093	0.098	**0.169**	0.085	0.096
D-EMA	Mistral-7B	0.313	0.103	0.307	0.229	0.082	0.244
	DeepSeek-R1	0.321	0.114	0.311	**0.241**	0.078	0.200
	Zephyr-7B-β	0.319	0.111	0.309	0.238	0.077	**0.399**
	LLaMA-13B	0.321	0.113	0.306	0.297	**0.092**	0.269
	Qwen2.5-0.5B	0.296	0.107	0.284	0.220	0.072	0.198
	smolLM2-1.7B	0.305	0.113	0.297	0.226	0.075	0.201
D-EMA + FM-Attn (c-EMA)	Mistral-7B	0.317	0.118	0.311	0.262	0.097	0.270
	DeepSeek-R1	0.325	0.124	0.316	**0.277**	0.101	0.228
	Zephyr-7B-β	0.326	0.118	0.314	0.274	0.091	0.242
	LLaMA-13B	0.329	0.117	0.313	0.291	**0.104**	**0.269**
	Qwen2.5-0.5B	0.325	0.114	0.307	0.270	0.086	0.217
	smolLM2-1.7B	0.326	0.116	0.314	0.278	0.096	0.244
C-MEGA	Mistral-7B	0.342	0.129	0.316	0.309	0.114	0.284
	DeepSeek-R1	**0.348**	**0.134**	**0.324**	**0.320**	**0.126**	**0.292**
	Zephyr-7B-β	0.344	0.131	0.319	0.316	0.123	0.289
	LLaMA-13B	0.337	0.128	0.321	0.319	0.124	0.291
	Qwen2.5-0.5B	0.327	0.124	0.315	0.297	0.108	0.257
	smolLM2-1.7B	0.343	0.128	0.322	0.321	0.121	0.289
C-MEGA (SBERT Seed)	Mistral-7B	0.238	0.088	0.234	0.241	0.091	0.205
	DeepSeek-R1	**0.244**	**0.093**	**0.241**	**0.251**	**0.098**	**0.212**
	Zephyr-7B-β	0.232	0.087	0.236	0.236	0.081	0.198
	LLaMA-13B	0.219	0.083	0.228	0.241	0.089	0.206
	Qwen2.5-0.5B	0.214	0.079	0.221	0.224	0.078	0.193
	smolLM2-1.7B	0.229	0.085	0.232	0.238	0.093	0.201
C-MEGA (Gold Summary Rollout)	Mistral-7B	0.676	0.524	0.604	0.612	0.452	0.503
	DeepSeek-R1	**0.710**	**0.543**	**0.627**	**0.632**	**0.473**	**0.524**
	Zephyr-7B-β	0.695	0.530	0.607	0.624	0.471	0.518
	LLaMA-13B	0.685	0.533	0.614	0.627	0.473	0.521
	Qwen2.5-0.5B	0.652	0.467	0.585	0.584	0.434	0.458
	smolLM2-1.7B	0.700	0.536	0.615	0.628	0.470	0.521

Table 12: **Performance comparison of LLMs and SLMs under 2-shot prompting and different `PerDucer` encoder variants.** *SLMs are not evaluated with user-history prompting due to limited context length.

Temperature	Model	PSE Scores		
		PSE-JSD	PSE-SU4	PSE-MET
0.2	Mistral-7B	0.342	0.129	0.316
	DeepSeek-R1	**0.348**	**0.134**	**0.324**
	Zephyr-7B-β	0.344	0.131	0.319
	LLaMA-13B	0.337	0.128	0.321
0.5	Mistral-7B	0.295	0.103	0.267
	DeepSeek-R1	0.318	0.117	0.304
	Zephyr-7B-β	0.302	0.095	0.256
	LLaMA-13B	0.289	0.113	0.260
0.8	Mistral-7B	0.255	0.078	0.187
	DeepSeek-R1	0.266	0.080	0.210
	Zephyr-7B-β	0.236	0.075	0.209
	LLaMA-13B	0.250	0.079	0.213

Table 13: **Effect of decoding temperature on performance across LLMs.** Lower temperatures consistently yield higher PSE scores across models.

Rollout length l	DeepSeek 2-shot			PerDucer+DeepSeek			Mistral 2-shot			PerDucer+Mistral		
	JSD	SU4	MET	JSD	SU4	MET	JSD	SU4	MET	JSD	SU4	MET
50	0.273	0.101	0.112	0.354	0.137	0.331	0.251	0.098	0.108	0.341	0.127	0.314
100	0.241	0.093	0.108	0.343	0.132	0.317	0.239	0.094	0.097	0.337	0.124	0.314
150	0.238	0.087	0.102	0.348	0.134	0.324	0.226	0.086	0.083	0.342	0.129	0.316

Table 14: **Rollout-length stability on PENS**. We report PSE-JSD (JSD), PSE-SU4 (SU4), and PSE-METEOR (MET) for 2-shot baselines and PerDucer-augmented LLMs over the first $n_q \in \{50, 100, 150\}$ queries.

Model (Sparse Click-only Test)	PSE-JSD	PSE-SU4	PSE-METEOR
DeepSeek (2-shot)	0.248/0.147 (–40.7%)	0.094/0.064 (–31.9%)	0.097/0.071 (–26.8%)
PerDucer+DeepSeek	**0.348/0.292** (–16.1%)	**0.134/0.108** (–19.4%)	**0.324/0.285** (–12.0%)
Mistral-7B (2-shot)	**0.226/0.117** (–48.2%)	**0.086/0.053** (–38.4%)	**0.083/0.048** (–42.2%)
PerDucer+Mistral-7B	**0.342/0.285** (–16.7%)	**0.129/0.103** (–20.2%)	**0.316/0.268** (–15.2%)

Table 15: **Sparse click-only stress test on PENS**. Each cell reports original performance / sparse-click performance, followed by the relative drop. The sparse-click projection retains only click interactions in the history.

Model	RMSD	HJ Rating
EBNR-1	0.932	2
EBNR-2	0.938	2
NAML-1	0.926	2
NRMS-1	0.911	2
NRMS-2	0.919	2
GTP	0.938	2
SP	0.881	3
Mistral (2-shot)	0.791	5
DeepSeek (2-shot)	0.779	5
PerDucer + DeepSeek	0.496	7
PerDucer + Mistral	0.542	7

Table 16: RMSD w.r.t. gold reference summaries and approximate HJ ratings.