

BEYOND SIMILARITY FOR PERSONALIZATION: USER MEMORY SELECTION VIA RESPONSE-UTILITY OPTIMIZATION

Anonymous authors

Paper under double-blind review

ABSTRACT

A common approach to personalization in large language models (LLMs) is to incorporate a subset of the user memory into the prompt at inference time to guide the model’s generation. Existing methods to select these subsets primarily rely on similarity between user memory items and input queries, ignoring how features actually affect the model’s predictive distribution. We propose **Response-Utility optimization for Memory Selection (RUMS)**, a novel user memory selection method, inspired by Bayesian Optimal Experimental Design, that directly quantifies how much each memory item reduces uncertainty in the model’s response distribution. RUMS measures mutual information between a subset of user memory and model outputs to identify items that sharpen predictions beyond semantic similarity. Even more, RUMS, by design, automatically selects if personalization is beneficial at all. We demonstrate that this information-theoretic foundation enables more principled user memory selection that aligns more closely with human selection compared to state-of-the-art methods, and models 400x bigger. Additionally, we show that memory items selected using RUMS result in better response quality compared to existing approaches, while having up to 95% reduction in cost.

1 INTRODUCTION

Personalization has long been recognized as a key component in effective human–computer interaction, and large language models (LLMs) are no exception. Users increasingly expect models to remember relevant details and adapt responses to their unique preferences. To meet this demand, current approaches to personalization generally fall into two categories: training-based methods and inference-time methods. Training-based approaches, such as low-rank preference models (Bose et al., 2025) or parameter-efficient fine-tuning modules (Tan et al., 2024), adapt the model parameters for individual users. Although these methods can be effective for small populations, they become computationally prohibitive when scaled to many users. As a result, inference-time personalization, such as including user-specific context in the input prompt, is often the preferred solution.

However, inference-time methods raise a central question: **What information, if any, should be included to guide the model?** A naïve strategy is to include all stored user memories in the prompt. However, this often reduces performance, as irrelevant or distracting information may overwhelm the model and produce responses that are less accurate or sound unnatural (Li et al., 2023; Rajeev et al., 2025). Moreover, this strategy becomes increasingly impractical as the memory size grows. Therefore, existing methods instead attempt to select and retrieve relevant subsets of the user memory by measuring the relationship between a *memory item and the input query*. In other words, these methods try to pick out only the user memory items that seem most related to the user’s current request. This is often done by heuristic retrieval (Lù, 2024), surface-level semantic similarity (Lei et al., 2023), or prompting LLMs (Handa et al., 2024). However, we argue that these approaches are fundamentally limited because they do not assess a *memory item’s direct impact on the model’s response* distribution. Instead, they rely on surface-level signals, making them more prone to missing important connections (under-selecting) or including superfluous ones (over-selecting). This can introduce noisy or irrelevant context that degrades response quality while also increasing latency, cost, and privacy risks (Hagström et al., 2025). These shortcomings highlight the need for lightweight, adaptable, response-aware methods that capture the true influence of user items on model outputs.

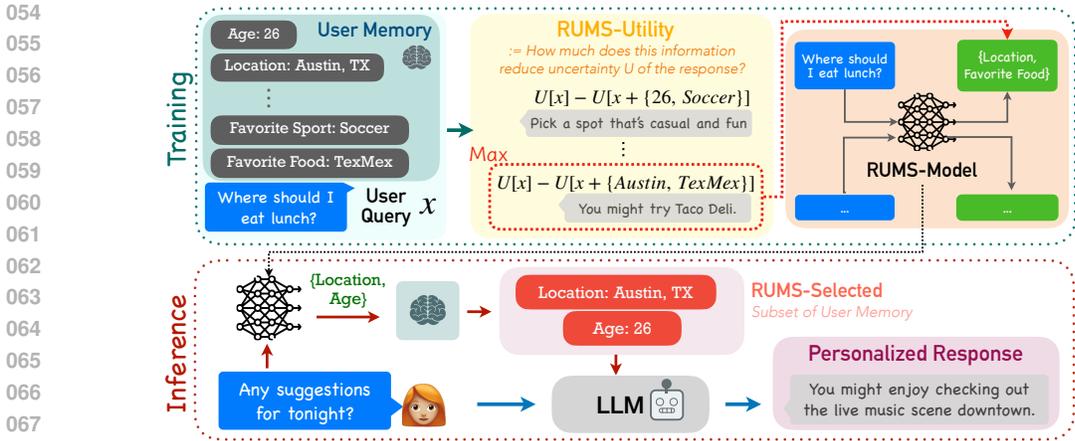


Figure 1: Overview of RUMS. RUMS-Utility quantifies how much memory subsets reduce response uncertainty (training phase), while RUMS-Models efficiently selects informative items at inference.

To bridge this gap, we introduce Response-Utility for Memory Selection (RUMS), which comprises of two key components: a novel utility function (*RUMS-Utility*) that quantifies the informativeness of user memory subsets, and a lightweight model (*RUMS-Model*) that efficiently approximates this utility computation at inference time.

RUMS-Utility draws inspiration from Bayesian Optimal Experimental Design (BOED) to directly measure how much each subset of user memories reduces uncertainty in the model’s response distribution. Specifically, it computes the conditional mutual information between candidate memory subsets and the model’s output distribution given the input query. This information-theoretic approach fundamentally differs from prior similarity-based methods, as rather than selecting memories that are semantically related to the query, it identifies those that meaningfully sharpen the model’s predictions. The utility function also naturally handles the decision of whether personalization should be applied at all: when no memory subset achieves sufficient utility (i.e., does not reduce response uncertainty beyond a threshold), the system forgoes personalization entirely.

However, computing *RUMS-Utility* for all queries is computationally expensive, particularly with large user profiles. Therefore, we use the utility function during an offline training phase to generate high-quality training data in the form (user query, optimal memory subset). We then train *RUMS-Models*, lightweight encoder-classifiers based on DeBERTa (He et al., 2021), to directly predict memory items that are most informative given a query. This learned approach preserves the principled selection quality of the utility function while enabling efficient deployment. We show that RUMS enables more principled memory item selection that is more closely aligned to human selection compared to state-of-the-art (SOTA) memory selection methods and models 400× its size. We also show that the items selected using RUMS result in better response quality than other methods, while lowering the cost of personalization overall.

Our work offers novel conceptual, theoretical, and empirical contributions to personalization:

- We propose RUMS, a lightweight, response-based method for user memory selection that directly accounts for the impact of memory items on model outputs.
- We introduce a principled utility function RUMS-Utility, designed to capture personalization benefit signal and can be used to support downstream training of a RUMS-Model.
- We show that RUMS (i) reliably identifies when personalization is beneficial, aligning closely with human judgments, (ii) selects memory subsets that align more strongly with human annotations than SOTA retrieval and prompting baselines, and (iii) improves generation quality over SOTA baselines and 400× larger models on synthetic and real-world datasets, while adapting to underlying model capabilities and costing up to 95% less than other methods.

2 RELATED WORKS

User Memory for LLM Personalization Memory augmentation enables personalization in LLMs by retrieving subsets of user memory at inference time to guide responses. Simple approaches prepend user context to prompts, such as profile summaries (Richardson et al., 2023; Pan et al.,

2025) or past interactions (Kang et al., 2023; Liu et al., 2023). However, as the amount of user memory grows, these methods quickly become intractable. Due to this, there has been an increased trend in selecting a *subset* of the user memory to prepend. For example, Chen (2025) retrieves relevant historical items via semantic similarity for recommendations, while LaMP (Salemi et al., 2023) investigates personalized sentence retrieval. PEARL (Mysore et al., 2024) and ROPG (Salemi et al., 2024) train retrieval components with user signals to improve personalization. For a broader overview, see Liu et al. (2025). However, these methods rely on the similarity between user input and memory item, without considering their impact on the model’s downstream response.

Selective Personalization A related line of research is selective conditioning or abstention which studies how models can ignore user memory or abstain from predictions when inputs are uninformative. Geifman & El-Yaniv (2017) introduce selective classification for neural networks, and recent work extends this to language models, proposing mechanisms to ignore input features or context that may degrade performance (Hagström et al., 2025). In memory selection, this corresponds to sometimes choosing no subset when personalization is unnecessary. Prior work typically relies on global thresholds (Li, 2023) or static policies (Joren et al., 2023), whereas our method learns a sequential, adaptive policy that dynamically selects which items to condition on based on the input and context.

Information-theoretic Uses in NLP Information-theoretic ideas have long informed NLP (Brown et al., 1992), spanning multimodal learning (Almudévar et al., 2025), information control (Ravfogel et al., 2020), interpretability (Chen et al., 2018), and prompt design (Zhang et al., 2022). In feature selection, EDDI (Ma et al., 2019) estimated conditional mutual information for individual features, while later work extended this to dynamic or sequential selection (Covert et al., 2023; Gadgil et al., 2024). Recent methods also use differences in next-token likelihood with and without context to guide generation through distribution editing (He et al., 2025). While much of this work has focused on domains such as image classification and fraud detection, the closest parallel to personalization arises in recommendation systems using Bayesian active learning to select a model query to ask the user, with the goal of improving recommendation quality (Handa et al., 2024). We draw on these principles to select memory items that maximize information gain for personalized responses.

3 METHOD

We aim to develop a method that selects a subset of user memory that maximizes the utility of the response generated with these items for the user.

Problem Formulation Consider user u with query x (e.g. “Where should I eat lunch?”). We let θ denote an LLM that generates an output $y = (y_1, \dots, y_T)$ of size T using distribution $p_\theta(y|x)$. We measure the quality of the response y by utility to the user $U_u(y)$. We consider a set of memory items \mathcal{M} (e.g. Location), in which user u has specific values m_u (e.g. Location: Austin, TX). Our aim is to select the memory subset $S \subseteq \mathcal{M}$ with instantiated values $m_{u,S} = \{m_u^{(a)} : a \in S\}$, which when given to the model with the input x yields the highest response utility. Formally,

$$S^* = \arg \max_{S \subseteq \mathcal{M}} \mathbb{E}_{y \sim p_\theta(\cdot|x, m_{u,S})} [U_u(y)]. \quad (1)$$

3.1 RUMS: RESPONSE UTILITY OPTIMIZED MEMORY SELECTION

Personalization is fundamentally an information acquisition problem: from a large space of user data, we must select the pieces that most improve, rather than distract from, the model’s response. A key insight, is that this parallels a well-studied problem in information-theoretic, Bayesian Optimal Experimental Design (BOED) which is choosing experiments that provide the most information about an unknown parameter. Both tasks involve selecting informative observations from a large space to maximize utility under uncertainty.

Formally, in BOED, for unknown parameter ϕ , design s , and potential outcome y , the optimal design is selected as $s^* = \arg \max_s \Omega(\phi; s, y)$, where Ω is the utility function. However, since the actual outcome y is unknown in advance, BOED instead maximizes the *expected* information gain (EIG):

$$\text{EIG}(s) := \mathbb{E}_{p(y|s)} [\Omega(\phi; s, y)] = \mathbb{E}_{p(y|s)} [H_\phi(y) - H_\phi(y|s)], \quad (2)$$

where H is the Shannon entropy (definition in Appendix D.1).

Memory selection can thus be reframed as a *personalization* design problem, where each subset of user memory items serves as an experimental design and the model’s response distribution as the outcome, transforming ad-hoc selection into a principled, theory-grounded optimization.

3.1.1 RUMS UTILITY FUNCTION

Applying BOED to LLM personalization poses three critical challenges. First, unlike classical BOED, which infers latent parameters, we aim to optimize the model’s predictive distribution directly, focusing on better responses rather than user preferences. Second, LLMs’ exponentially large, variable-length output spaces makes standard BOED computations intractable. Third, we require a method which also detects when personalization truly improves responses rather than adding noise. Our approach addresses all three through a novel BOED adaptation.

Challenge 1: From parameter inference to distribution shaping. Classical BOED methods aim to reduce uncertainty about an unknown parameter ϕ through experimental design s (Equation (2)). In our setting, however, there is no latent parameter to infer. Instead, we want to shape the model’s distribution over responses y to be more precise and relevant. We address this difference by introducing a novel utility function based on the reduction in predictive entropy:

$$\text{EIG}(S, u) = \mathbb{E}_{p_\theta(y|x)} [H_\theta(y | x)] - \mathbb{E}_{p_\theta(y|x, m_{u,S})} [H_\theta(y | x, m_{u,S})] \quad (3)$$

we use H_θ to be the Shannon entropy over the output distribution using the model θ .

The key idea is that a memory subset S is valuable to the extent it reduces uncertainty in the model’s predictions for user u . For instance, for the query “where should I eat lunch?”, without user information, the model spreads probability across countless plausible restaurants, yielding high entropy. However, adding even a single relevant fact, such as “Location: Austin, TX” or “Dietary preference: Vegan”, sharply concentrates probability on relevant options. Therefore, by measuring this entropy reduction, the utility function directly quantifies how much S sharpens predictions.

We note that some memory items are highly dependent on specific user values (e.g., Dietary Preference: Vegan vs. None), while others are informative regardless of value (e.g., Location). When the exact user is known, this information is applied directly; when unknown, averaging over many user profiles allows the method to transition from user-specific to user-agnostic.

Challenge 2: From intractable to tractable. However, when using LLMs, directly computing equation 3 is intractable because y is a sequence of length T , and the entropy requires summing over the exponentially many possible outputs. We address this issue through a key theoretical insight.

Proposition 1. *The entropy of the sequence distribution can be decomposed as the sum of conditional token entropies. Given $p_\theta(\cdot | y_{<t}, x)$, the conditional next-token distribution given prefix $y_{<t}$,*

$$H_\theta(y | x) := \mathbb{E}_{y \sim p_\theta(\cdot|x)} [-\log P_\theta(y | x)] = \mathbb{E}_{y_{<t} \sim p_\theta(\cdot|x)} \left[\sum_{t=1}^T H(p_\theta(y_t | y_{<t}, x)) \right].$$

This follows directly from the chain rule of probability and the law of iterated expectations (proof in Appendix E).

Although Proposition 1 reduces the problem to token-level entropies, the outer expectation over prefixes $y_{<t}$ remains intractable. We therefore employ Monte Carlo sampling, generating N sequences and computing:

$$\hat{H}_\theta(y | x) = \frac{1}{N} \sum_{i=1}^N \left[\frac{1}{T} \sum_{t=1}^T H(p_\theta(y_t^i | y_{<t}^i, x)) \right]. \quad (4)$$

Here, $y^i = (y_1^i, \dots, y_T^i)$ denotes the i -th sampled sequence. We note that $T \equiv T_N$ if token sizes is not fixed per sample. This approximation converges to the true per-token sequence entropy as $N \rightarrow \infty$ under i.i.d. sampling from $p_\theta(\cdot | x)$ with finite expectation. In practice, however, sequence samples are not strictly i.i.d., and factors such as constrained decoding or approximate distributions can introduce bias. Despite these limitations, we find that using $N = 5$ samples provides a good balance between accuracy and efficiency. For more empirical analysis, see Appendix B.1.

An analogous estimate is computed for $\hat{H}_\theta(y | x, m_{u,S})$ in order to approximate Equation (3). Together, we use the estimated EIG for personalization,

$$\widehat{\text{EIG}}(S, u) := [\hat{H}_\theta(y|x) - \hat{H}_\theta(y|x, m_{u,S})] \quad (5)$$

Then, our final objective selects the memory subset that maximizes this utility:

$$S^*(x, u) = \arg \max_{S \subseteq M} \widehat{EIG}(S, u). \quad (6)$$

This provides an information-theoretic approach to memory selection that directly measures impact on response quality. For clarity, we define the maximum utility score for a given input x and user u as RUMS-Utility and the subset of user memory items selected by Equation (6) as RUMS-Subset.

Challenge 3: Automatic detection of when personalization helps. Unlike prior methods that always retrieve information, our formulation naturally identifies when personalization is unnecessary. When no memory subset achieves sufficient entropy reduction (i.e. $\widehat{EIG}(S) < \tau$ for threshold τ), the system selects the empty set, preventing irrelevant information from degrading response quality. This built-in selectivity eliminates the need for separate pre-filtering mechanisms, reducing computational overhead in deployment.

3.1.2 RUMS-MODELS: LEARNING TO APPROXIMATE UTILITY

While RUMS offers principled memory selection, computing entropy reductions for all possible subsets at inference is prohibitive, especially with large user profiles. We address this via amortized learning, where RUMS generates high-quality training data offline, and a lightweight neural model is trained to predict optimal memory selections directly at inference.

Training Data Generation Our training pipeline uses RUMS to generate (user query, optimal memory subset) pairs, with a threshold β determining whether a subset provides enough entropy reduction to be selected.

1. *Profile Generation and Candidate Reduction:* RUMS relies on how memory item *values* sharpen response distributions, so a set of plausible user profiles u is needed to estimate utility. When exact profiles are unavailable or a more general framework is desired, a diverse population-like set can be generated or collected. Optionally, for very large profiles (e.g., ≥ 50 items), a larger LLM can first filter them to a smaller candidate set of k items.
2. *Utility Computation:* For each candidate subset of memory, we compute the estimated utility score $\widehat{EIG}(S, u)$ given a random user profile u . To capture interaction effects between memory items, we evaluate subsets rather than individual items, revealing more utility than the sum of their individual contributions.
3. *Optimal Memory Subset Selection:* We select the subset with maximum utility above threshold β , or optionally the empty set if none qualify. Tuned on validation data, β enables automatic abstention from personalization when it would not improve response quality (Appendix G.5).

Critically, we pair each query with many diverse user profiles, yielding a richer training set that supports user-agnostic selection policies. By observing how the same query benefits from different memory items depending on specific user values, the model learns to identify genuinely informative items rather than spurious correlations.

Model Architecture and Training We train RUMS-Models as lightweight classifiers to approximate the costly utility computation. For dynamic memory schemas, where item types vary across users, a binary classifier $f(x, m_i) \rightarrow \{0, 1\}$ evaluates each memory item $\{m_i\}_{i=1}^n$ independently, handling arbitrary structures but requiring n forward passes. For static schemas with d fixed item types, a multiclass classifier $f(x) \rightarrow \{0, 1\}^d$ outputs all selections in a single pass, preferred for efficiency. The resulting RUMS-Model matches the inference speed of simple similarity-based retrieval while retaining principled selection quality, making it practical for large-scale deployment.

4 RESULTS

In this section, we evaluate RUMS, our proposed response-utility method for memory selection. For clarity, we define three key outputs from RUMS. First, *RUMS-Utility* represents the maximum utility score computed using Equation (5). Second, *RUMS-Subset* is the subset of memory items chosen according to Equation (6). Finally, *RUMS-Models* are lightweight models trained on the data generated by RUMS. We evaluate our approach along three dimensions:

- H1: The utility scores from RUMS-Utility provides a reliable signal for distinguishing between inputs that benefit from personalization and those that do not.

H2: The subsets selected, RUMS-Subset, contain more informative items than state-of-the-art baselines and larger models. Furthermore, the utility signal is strong enough that models trained on them (RUMS-Models) also outperform baselines and larger models.

H3: Memory subsets selected with RUMS-Models improve downstream response generation compared to state-of-the-art baselines and larger models.

4.1 EXPERIMENTAL SETUP

Across all experiments (H1–H3), we consider $n = 50$ static user memory items (e.g., *location*, *age*, *name*), and assumes that all values are given for each user (see Appendix G.3 for list of items). The task is to select up to 5 items that are most relevant for answering a given user input x .

Datasets. To test both instances that would and would not benefit from personalization, we use a diverse mixture of possible user inputs. We evaluate across 3 datasets:

- **PersonaFeedback** (Tao et al., 2025): A synthetic dataset of user inputs generated by an LLM using randomly sampled, unique user memory data and scenarios. This process ensured that each input relied on personal details and would benefit from personalization. The original data is in Chinese, so we translated with a translation model (Tiedemann & Thottingal, 2020). Example: “Do you have any music for the trip?”
- **FreebaseQA** (Jiang et al., 2019): A dataset of trivia-style questions and answers. Example: “What type of snake is a sidewinder?”
- **WildChat** (Zhao et al., 2024): Real-world ChatGPT interactions (English only, non-toxic, and request-based). We use only the first human-written input. Examples: “im hungry”; “who is Mohamed Gamal Mohamed Maniieh”

By nature of the data, we classify PersonaFeedback as personalized inputs and FreebaseQA as non-personalized inputs, and refer to these together as the “Synthetic” dataset. For WildChat, two annotators labeled each input as personalized or non-personalized (agreement required). We denote these subsets as WildChat-Personal and WildChat-NonPersonal, which form the “Real-World” dataset.

User Profiles. User profiles are required for utility computation and evaluation. We first prompted GPT-4 (OpenAI et al., 2024) to generate 50 common user memory items. We then used the model again to generate consistent user profiles without contradictions (e.g., “vegan” with “favorite food = steak”). We generated $n = 100$ unique profiles each for training and testing. Details on prompts used, distributions of memory item values and generation process are provided in Appendix G.3.

RUMS-Model Training To create training data with RUMS, we used Llama-3.1-8B (AI, 2024b) to compute the entropy. Following the procedure in Section 3.1.2, GPT-4 was first prompted to reduce the candidate memory items from 50 to at most 10 for each query before computing the utility scores. The utility scores were estimated with $n = 5$ Monte Carlo samples. Lastly, a threshold of $t = 0.29$, determined on a held-out validation set, was applied to choose between the subset with maximum utility or the empty set (i.e., no personalization). We use $n = 2.5K$ from PersonaFeedback and 2K from FreebaseQA for training. Since both datasets are also used for evaluation, we rely on held-out splits to avoid overlap and interpret results as in-domain performance. To assess generalization, we also evaluate on WildChat, a real-world dataset from a different distribution.

We train two RUMS-Model variants: The first, RUMS-Binary, is a binary classifier trained on individual (query, memory value) pairs. Because each memory value is treated separately and negative labels dominate, this model tends to under-select, making it better suited for noise-sensitive settings. The second, RUMS-Multi, is a multiclass classifier trained on denser labels constructed by merging the selected subsets from the $u = 5$ user profiles per query. In this setup, all memory items chosen using any of the five profiles were marked as significant, encouraging broader selection. As a result, the model tends to over-select, which is advantageous for more robust models that can filter noise. Both are based on DeBERTa-v3-large (He et al., 2021). Full details are in Appendix G.5.

Baselines Current approaches to memory selection primarily rely on two strategies: semantic similarity-based retrieval and large language model prompting. To comprehensively evaluate against existing methods, we employ baselines representing both approaches: (1) **Semantic Similarity:** We compute cosine similarity between input and memory item embeddings using a sentence transformer (Wang et al., 2020), representing the dominant paradigm of surface-level similarity matching in cur-

Table 1: Recall (accuracy on personalized) and specificity (accuracy on non-personalized) comparison across datasets and methods. **Bolded** is the highest value in each category.

Dataset	Recall (Personalized)				Specificity (Non-Personalized)			
	RUMS-Utility	GPT4-Few	GPT4-Zero	Semantic Similarity	RUMS-Utility	GPT4-Few	GPT4-Zero	Semantic Similarity
Synthetic (n=20)	94.4	100.0	33.3	100.0	94.4	50.0	50.0	0.0
Real-World (n=31)	92.3	76.9	53.9	7.7	96.0	100.0	100.0	100.0

rent retrieval systems, (2) **GPT-4 Prompting**: We use direct prompting of GPT-4 (*zero-shot* and *few-shot* - up to 4 in-context examples) as a representative LLM-based approach and upper bound on model capability, given its 400× larger size compared to RUMS-Models. For H3, which evaluates downstream generation quality, we thus additionally include established retrieval methods and naive baselines to ensure comprehensive coverage: (3) **SOTA Retrieval Methods**: We evaluate *BM25* (Lù, 2024) and *ReContriever* (Lei et al., 2023) as representative SOTA retrieval approaches for memory selection, (4) **Naive Baselines**: We include *None* (no items), *All* (50 items), and *Random* (5 items) to establish performance bounds and test robustness to different selection strategies.

4.2 H1: DOES RUMS-UTILITY DISTINGUISH PERSONALIZED FROM NON-PERSONALIZED INPUTS?

We hypothesize that entropy reduction when conditioning on user memory reflects its importance. Thus, inputs that do not benefit from personalization should yield low maximum utility scores (i.e., the maximum utility score across all subsets of candidate memory items for a specific user), while personalized inputs should yield high ones. To test this, we compare the maximum utility scores on a known non-personalized (FreebaseQA) and a known personalized (PersonaFeedback) datasets. We sampled 500 inputs per dataset and computed the maximum utility score using Equation (5) for each input. The resulting distributions were compared using divergence metrics and hypothesis testing.

Utility scores separate personalized from non-personalized inputs. In Figure 2 we show that personalized inputs exhibit much higher utility scores (mean = 0.50, std = 0.12) than non-personalized ones (mean = 0.12, std = 0.09). The Jensen-Shannon divergence between distributions is high at 0.66, and a two-sample Kolmogorov-Smirnov test yields a statistic of 0.82 ($p < .001$), strongly rejecting the null hypothesis of equal distributions. These results indicate that maximum utility scores reliably separate personalized from non-personalized inputs. For further exploration on hyperparameter selection and its effects on the robustness of these results see Appendix B.1.

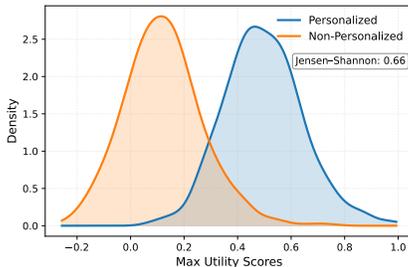


Figure 2: Distribution of maximum utility scores for personalized (PersonaFeedback) and non-personalized (FreebaseQA) inputs.

4.2.1 H1': DO UTILITY SCORES ALIGN WITH HUMAN JUDGMENTS OF PERSONALIZATION?

We next test whether personalization decisions made by utility score thresholding aligns with human judgment. We randomly sampled $n = 25$ inputs from each of PersonaFeedback, FreebaseQA, WildChat-NonPersonal, and WildChat-Personal. Four human annotators labeled whether each input benefits from personalization, which yielded an agreement between the parties of Cohen's $\kappa = 0.59$; we retained only inputs with unanimous agreement ($n = 64$).

In order to get a robust signal using RUMS, we used $n = 10$ Monte Carlo samples and $u = 10$ randomly selected user-profiles. We used the average maximum utility score over the profiles as RUMS-Utility. We then tuned a threshold on 20% validation data. We compared RUMS-Utility against GPT-4 (prompted to determine personalization necessity) and semantic similarity baselines, and the performance was measured by recall (accuracy in identifying personalized inputs) and specificity (accuracy in identifying non-personalized inputs). If both humans and the model selected no items, the sample was counted correct for both metrics.

Table 2: Precision, recall, and F1 scores of memory items selected compared to human gold label annotation across datasets and methods.

Dataset	Metric	RUMS-Subset	RUMS-Binary	RUMS-Multi	Semantic Sim.	GPT4-Zero	GPT4-Few
Synthetic	Precision	0.61	0.60	0.23	0.09	0.26	0.53
	Recall	0.70	0.89	0.39	0.23	0.42	0.65
	F1	0.70	0.78	0.31	0.14	0.35	0.64
Real World	Precision	0.48	0.32	0.30	0.17	0.39	0.42
	Recall	0.36	0.60	0.38	0.27	0.52	0.39
	F1	0.53	0.45	0.35	0.20	0.43	0.39

Table 3: Average win rate of our method against baselines on Synthetic (left) and Real World (right) datasets. Cells in green mark cases where our method outperforms the baseline, and cells in red mark cases where the baseline does better. Values with magnitude above 0.1 are bolded to highlight strong differences. Results are reported for two response models (LLaMA 70B Instruct and GPT-4) and two RUMS variants: RUMS-Binary (fewer items) and RUMS-Multi (more items).

	LLaMA 70B		GPT-4			LLaMA 70B		GPT-4	
	Binary	Multi	Binary	Multi		Binary	Multi	Binary	Multi
None	0.25	-0.13	0.26	0.36	None	0.05	0.03	0.20	0.17
All	0.16	-0.19	-0.16	0.02	All	-0.04	-0.10	-0.14	-0.06
Random	0.34	0.13	0.06	0.19	Random	0.11	0.13	0.10	0.04
Semantic Sim.	0.27	0.09	-0.03	0.05	Semantic Sim.	-0.03	0.00	-0.02	0.00
BM25	0.19	-0.13	0.20	0.32	BM25	0.01	0.05	0.18	0.13
ReContriever	0.27	-0.03	-0.04	0.03	ReContriever	0.06	0.01	-0.05	0.13
GPT4-Few	0.18	-0.16	-0.09	-0.01	GPT4-Few	0.00	-0.08	0.04	0.11

RUMS-Utility correlate more strongly with human judgments. As shown in Table 1, RUMS-Utility outperforms GPT-4 and semantic similarity baselines. Compared to GPT-4 FewShot, it achieves +7% recall and +15% specificity overall. Gains are strongest on synthetic data, with competitive results on real-world. Other baselines show imbalanced behavior (e.g., GPT-4 ZeroShot often avoids personalization, giving high specificity but low recall). In contrast, utility scores perform well on both metrics, reflecting their ability to capture each item’s true effect on generation.

4.3 H2: DOES RUMS-SUBSET BETTER MATCH HUMAN JUDGMENT?

Building on H1, we now ask whether our method identifies which specific memory items help personalization. We randomly selected $n = 15$ samples from each of PersonaFeedback, FreebaseQA, WildChat-NonPersonal, and WildChat-Personal for a total of $n = 60$. As before, four annotators selected any of 50 possible items (including “No Items”) they judged useful, yielding 41% majority ($k \geq 3$) agreement. While moderate, this level of agreement is expected given the subjective nature of the task. We therefore considered an item as selected by humans if a majority agreed, resulting in an average of 4.19 items selected per user query for inputs where at least one item was chosen.

We evaluated (i) direct selection from utility scores RUMS-Subset (up to 5) and (ii) RUMS-Models, the trained models described in Section 4. We compare RUMS-Subset and RUMS-Models against GPT-4 (prompted to select up to five relevant memory items, both zero- and few-shot) and semantic similarity (selecting up to five highest-scoring items with cosine similarity > 0). Performance was measured with precision, recall, and F1 against human labels.

RUMS-Subset aligns more closely with human annotation. As shown in Table 2, the two methods based on RUMS achieve on average 20% higher precision, 31% higher recall, and 9% higher F1 compared to the baselines across both synthetic and real-world datasets. Semantic similarity performs worst, reflecting reliance on surface overlap rather than downstream utility. GPT-4 shows strong recall but substantially lower precision, indicating that it tends to over-select. In contrast, the results show that the raw utility scores, RUMS-Subset, themselves provide strong signal for identifying useful memory items, and RUMS-Models achieves similar F1, demonstrating that small models with only modest amounts of data can learn effective memory selection from utility scores. For examples of memory items selected see Appendix B.3

4.4 H3: DO ITEMS SELECTED BY RUMS-MODELS IMPROVE DOWNSTREAM GENERATION?

Having established that RUMS makes effective decisions about when and what to personalize (H1-H2), we now demonstrate that these selections translate to tangible improvements in response quality. This tests our core claim: principled selection leads to better personalization.

Experimental Setup. We evaluate RUMS on both synthetic (PersonaFeedback and FreebaseQA, 100 samples each) and real-world datasets (WildChat, 50 personalized and 50 non-personalized) to assess generalization. Responses are generated using two models: GPT-4, a high-performing model with robust capabilities, and LLaMA 70B Instruct (AI, 2024a), a smaller open-source model that is relatively more susceptible to noise. We evaluate response quality using RUMS-Models against all baselines described in Section 4.1. Performance is measured by GPT-4 win rate using chain-of-thought prompting and confidence thresholding (Appendix G.7).

RUMS improves response quality across diverse settings

Table 3 reports generation win rates comparing our method against the baselines synthetic and real-world datasets. On the synthetic dataset, RUMS-Binary consistently outperforms all baselines, achieving at least a 15% higher win rate when responses are generated using the less robust LLaMA 70B model. Most notably, it has an 18% winrate over GPT4-Fewshot, a model 400× its size. Most of these gains come from non-personalized inputs, where a lower-capability model struggles to ignore noisy items, reducing its performance. In contrast, when using a more capable model such as GPT-4 for response generation, RUMS-Multi outperforms almost all baselines, primarily driven by personalized inputs. In this setting, the model can leverage more items, including potentially noisy ones, while still filtering out unhelpful items to a large extent. Real-world results follow the same trend. RUMS-Binary achieves a 12% higher win rate than the best SOTA baseline, ReContriever with LLaMA 70B, while RUMS-Multi exceeds GPT4-FewShot by 11% and BM25/ReContriever by 13% with GPT-4. The *All* baseline performs reasonably with only 50 items, but prior work shows scaling degrades performance (Rajeev et al., 2025). See Appendix B.5 for breakdowns.

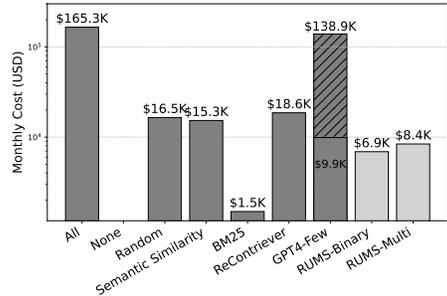
RUMS-Models reduces cost compared to other under-performing methods. Because personalization may be applied to every model input, efficiency is critical. We track both the number of memory items selected and the estimated monthly inference cost, computed as the extra tokens from selected items multiplied by a pricing model of \$0.01 per 1K tokens for one million daily queries (current GPT-4 rates). Figure 3a shows that RUMS-Models select substantially fewer memory items than most baselines, translating to significant cost savings (Figure 3b): RUMS-Binary costs \$6.9K/month and RUMS-Multi costs \$8.4K/month, compared to \$165.3K for *All* and \$18.6K for the competitive ReContriever. Both RUMS variants are cheaper than GPT4-FewShot, even when the additional cost of memory subset selection required by GPT4 baselines is excluded. Thus, RUMS balances informativeness and efficiency, enabling scalable personalization.

5 CONCLUSION

We propose RUMS, a new method for memory subset selection grounded in information theory that provides a lightweight, adaptable solution for large-scale personalization. Through extensive experimentation, we demonstrate that RUMS reliably determines when personalization is beneficial, selects memory items that align with human preferences, and improves response generation quality. Importantly, RUMS achieves these results at up to 95% lower computational cost than existing methods, enabling scalable large-scale personalization.

Method	Synthetic	Real World
All	50.00	50.00
None	0.00	0.00
Random	5.00	5.00
Semantic Similarity	4.63	4.79
BM25	0.60	0.35
ReContriever	5.00	5.00
GPT4-Few	2.80	3.00
RUMS-Binary	1.58	2.22

(a) Average number of memory items by method and dataset type.



(b) Estimated additional average monthly cost by method. Diagonal line indicate added cost of LLM calls for memory selection.

Figure 3: Cost Analysis

486 6 ETHICS

487
488 We believe this research presents minimal ethical concerns. However, like other methods that steer
489 or guide model generations, our approach could potentially introduce unintentional biases, reinforce
490 stereotypes, or produce harmful outputs. We therefore recommend that practitioners implement
491 comprehensive evaluation protocols before deployment, including stress testing, automated safety
492 evaluations, human assessments, and ongoing monitoring systems. Additionally, since our method
493 incorporates user information during inference, it introduces potential privacy and security vulnera-
494 bilities that require careful consideration in production environments.

495 7 REPRODUCIBILITY

496 To ensure reproducibility, we provide comprehensive supporting materials alongside this work. A
497 complete proof of our main proposition is available in Appendix E, and we will release all code
498 for implementing RUMS and reproducing our experiments via a public GitHub repository. De-
499 tailed experimental specifications are provided in Appendix G, including dataset descriptions (Ap-
500 pendix G.2), profile generation procedures (Appendix G.3), baseline configurations and hyperpa-
501 rameters (Appendix G.4), our method’s hyperparameters (Appendix G.5), evaluation protocols (Ap-
502 pendix G.7), and software/hardware specifications (Appendix H).

503 REFERENCES

- 504 Meta AI. Llama 3.1 70b instruct, 2024a. URL <https://huggingface.co/meta-llama/Llama-3.1-70B-Instruct>. Accessed: 2025-09-16.
- 505 Meta AI. Llama-3.1-8b, 2024b. URL <https://huggingface.co/meta-llama/Llama-3.1-8B>. Accessed: 2025-09-16.
- 506 Antonio Almudévar, Jose Miguel Hernandez-Lobato, Sameer Khurana, Ricard Marxer, and Alfonso
507 Ortega. Aligning multimodal representations through an information bottleneck. In *Proceedings*
508 *of the 2025 International Conference on Machine Learning (ICML 2025), Poster Session*, 2025.
509 URL <https://icml.cc/virtual/2025/poster/43456>.
- 510 José M. Bernardo. Expected information as expected utility. *The Annals of Statistics*, 7(3):
511 686–690, 1979. doi: 10.1214/aos/1176344689. URL [https://doi.org/10.1214/aos/](https://doi.org/10.1214/aos/1176344689)
512 [1176344689](https://doi.org/10.1214/aos/1176344689).
- 513 Avinandan Bose, Zhihan Xiong, Yuejie Chi, Simon Shaolei Du, Lin Xiao, and Maryam Fazel. Lore:
514 Personalizing llms via low-rank reward modeling, 2025. URL [https://arxiv.org/abs/](https://arxiv.org/abs/2504.14439)
515 [2504.14439](https://arxiv.org/abs/2504.14439).
- 516 Peter F. Brown, Stephen A. Della Pietra, Vincent J. Della Pietra, Jennifer C. Lai, and Robert L.
517 Mercer. An estimate of an upper bound for the entropy of English. *Computational Linguistics*,
518 18(1):31–40, 1992. URL <https://aclanthology.org/J92-1002/>.
- 519 Kathryn Chaloner and Isabella Verdinelli. Bayesian experimental design: A review. *Statistical*
520 *Science*, 10(3):273–304, 1995. doi: 10.1214/ss/1177009939. URL [https://www.jstor.](https://www.jstor.org/stable/2246015)
521 [org/stable/2246015](https://www.jstor.org/stable/2246015). Also available as JSTOR Stable URL.
- 522 Jianbo Chen, Le Song, Martin Wainwright, and Michael Jordan. Learning to explain: An
523 information-theoretic perspective on model interpretation. In Jennifer Dy and Andreas Krause
524 (eds.), *Proceedings of the 35th International Conference on Machine Learning (ICML 2018)*,
525 volume 80 of *Proceedings of Machine Learning Research*, pp. 883–892. PMLR, 10–15 Jul 2018.
526 URL <https://proceedings.mlr.press/v80/chen18j.html>.
- 527 Jiarui Chen. In *Memory Assisted LLM for Personalized Recommendation System*, 05 2025. doi:
528 10.48550/arXiv.2505.03824.
- 529 Ian Covert, Wei Qiu, Mingyu Lu, Nayoon Kim, Nathan White, and Su-In Lee. Learning to maxi-
530 mize mutual information for dynamic feature selection. In *Proceedings of the 40th International*
531 *Conference on Machine Learning, ICML’23*. JMLR.org, 2023.

- 540 Yijiang River Dong, Tiancheng Hu, and Nigel Collier. Can LLM be a personalized judge? In
541 Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Findings of the Association for*
542 *Computational Linguistics: EMNLP 2024*, pp. 10126–10141, Miami, Florida, USA, November
543 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-emnlp.592.
544 URL <https://aclanthology.org/2024.findings-emnlp.592/>.
- 545 Adam Foster, Martin Jankowiak, Eli Bingham, Paul Horsfall, Yee Whye Teh, Tom Rainforth, and
546 Noah Goodman. Variational bayesian optimal experimental design. 2019.
- 547 Soham Gadgil, Ian Covert, and Su-In Lee. Estimating conditional mutual information for dy-
548 namic feature selection. In *Proceedings of the 42nd International Conference on Learning Rep-*
549 *resentations (ICLR)*. OpenReview.net, 2024. URL [https://openreview.net/pdf?id=](https://openreview.net/pdf?id=Oju2Qu9jvn)
550 [Oju2Qu9jvn](https://openreview.net/pdf?id=Oju2Qu9jvn).
- 551 Yonatan Geifman and Ran El-Yaniv. Selective classification for deep neural networks. In *Proceed-*
552 *ings of the 31st International Conference on Neural Information Processing Systems, NIPS’17*,
553 pp. 4885–4894, Red Hook, NY, USA, 2017. Curran Associates Inc. ISBN 9781510860964.
- 554 Lovisa Hagström, Youna Kim, Haeun Yu, Sang goo Lee, Richard Johansson, Hyunsoo Cho, and
555 Isabelle Augenstein. Cub: Benchmarking context utilisation techniques for language models,
556 2025. URL <https://arxiv.org/abs/2505.16518>.
- 557 Kunal Handa, Yarin Gal, Ellie Pavlick, Noah Goodman, Jacob Andreas, Alex Tamkin, and Be-
558 linda Z. Li. Bayesian preference elicitation with language models, 2024. URL <https://arxiv.org/abs/2403.05534>.
- 559 Jerry Zhi-Yang He, Sashrika Pandey, Mariah L. Schrum, and Anca Dragan. Context steering: Con-
560 trollable personalization at inference time. In *Proceedings of the 13th International Conference on*
561 *Learning Representations (ICLR)*, 2025. URL [https://openreview.net/forum?id=](https://openreview.net/forum?id=xQCXInDq0m)
562 [xQCXInDq0m](https://openreview.net/forum?id=xQCXInDq0m).
- 563 Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. Deberta: Decoding-enhanced bert
564 with disentangled attention. In *International Conference on Learning Representations*, 2021.
565 URL <https://openreview.net/forum?id=XPZJaotutsD>.
- 566 Kelvin Jiang, Dekun Wu, and Hui Jiang. FreebaseQA: A new factoid QA data set matching trivia-
567 style question-answer pairs with Freebase. In Jill Burstein, Christy Doran, and Tamar Solorio
568 (eds.), *Proceedings of the 2019 Conference of the North American Chapter of the Association for*
569 *Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*,
570 pp. 318–323, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.
571 doi: 10.18653/v1/N19-1028. URL <https://aclanthology.org/N19-1028/>.
- 572 Hailey Joren, Chirag Nagpal, Katherine A Heller, and Berk Ustun. Participatory
573 personalization in classification. In A. Oh, T. Naumann, A. Globerson,
574 K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Pro-*
575 *cessing Systems*, volume 36, pp. 14113–14133. Curran Associates, Inc., 2023. URL
576 [https://proceedings.neurips.cc/paper_files/paper/2023/file/](https://proceedings.neurips.cc/paper_files/paper/2023/file/2dbb8bfe4cd3875609b23799830ee865-Paper-Conference.pdf)
577 [2dbb8bfe4cd3875609b23799830ee865-Paper-Conference.pdf](https://proceedings.neurips.cc/paper_files/paper/2023/file/2dbb8bfe4cd3875609b23799830ee865-Paper-Conference.pdf).
- 578 Wang-Cheng Kang, Jianmo Ni, Nikhil Mehta, Maheswaran Sathiamoorthy, Lichan Hong, Ed Chi,
579 and Derek Cheng. Do llms understand user preferences? evaluating llms on user rating prediction.
580 05 2023. doi: 10.48550/arXiv.2305.06474.
- 581 Andreas Köpf, Yannic Kilcher, Dimitri von Rütte, Sotiris Anagnostidis, Zhi Rui Tam, Keith Stevens,
582 Abdullah Barhoum, Duc Nguyen, Oliver Stanley, Richárd Nagyfi, Shahul ES, Sameer Suri,
583 David Glushkov, Arnav Dantuluri, Andrew Maguire, Christoph Schuhmann, Huu Nguyen, and
584 Alexander Mattick. Openassistant conversations: Democratizing large language model alignment.
585 In *NeurIPS 2023 Poster*, 2023. URL [https://neurips.cc/virtual/2023/poster/](https://neurips.cc/virtual/2023/poster/73573)
586 [73573](https://neurips.cc/virtual/2023/poster/73573).
- 587 Yibin Lei, Liang Ding, Yu Cao, Changtong Zan, Andrew Yates, and Dacheng Tao. Unsupervised
588 dense retrieval with relevance-aware contrastive pre-training. In Anna Rogers, Jordan Boyd-
589 Graber, and Naoaki Okazaki (eds.), *Findings of the Association for Computational Linguistics:*
590
591
592
593

- 594 *ACL 2023*, pp. 10932–10940, Toronto, Canada, July 2023. Association for Computational Lin-
 595 guistics. doi: 10.18653/v1/2023.findings-acl.695. URL [https://aclanthology.org/
 596 2023.findings-acl.695/](https://aclanthology.org/2023.findings-acl.695/).
 597
- 598 Yucheng Li. Unlocking context constraints of llms: Enhancing context efficiency of llms with self-
 599 information-based content filtering. In *Proceedings of the 2023 Conference on Empirical Methods
 600 in Natural Language Processing (EMNLP 2023)*, 2023. URL [https://arxiv.org/abs/
 601 2304.12102](https://arxiv.org/abs/2304.12102).
- 602 Yucheng Li, Dan Liang, Shuo Zhang, Chengyu Yin, Siyu Xu, Fandong Yue, Guoyang Zhou, Yiwei
 603 Wei, Huabin Xu, Yujie Wang, and Jian-Yun Zeng. Compressing context to enhance inference
 604 efficiency of large language models. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.),
 605 *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*,
 606 pp. 6348–6360, Singapore, December 2023. Association for Computational Linguistics. URL
 607 <https://aclanthology.org/2023.emnlp-main.391>.
- 608 Dennis V. Lindley. On a measure of the information provided by an experiment. *The Annals of
 609 Mathematical Statistics*, 27(4):986–1005, 1956. doi: 10.1214/aoms/1177728069. URL [https://
 610 doi.org/10.1214/aoms/1177728069](https://doi.org/10.1214/aoms/1177728069).
 611
- 612 Jiahong Liu, Zexuan Qiu, Zhongyang Li, Quanyu Dai, Jieming Zhu, Minda Hu, Menglin Yang, and
 613 Irwin King. A survey of personalized large language models: Progress and future directions. 02
 614 2025. doi: 10.48550/arXiv.2502.11528.
- 615 Junling Liu, Chao Liu, Peilin Zhou, Renjie Lv, Kang Zhou, and Yan Zhang. Is chatgpt a good
 616 recommender? a preliminary study, 2023. URL <https://arxiv.org/abs/2304.10149>.
 617
- 618 Xing Han Lù. Bm25s: Orders of magnitude faster lexical search via eager sparse scoring, 2024.
 619 URL <https://arxiv.org/abs/2407.03618>.
 620
- 621 Chao Ma, Sebastian Tschiatschek, Konstantina Palla, José Miguel Hernández-Lobato, Sebastian
 622 Nowozin, and Cheng Zhang. Eddi: Efficient dynamic discovery of high-value information with
 623 partial vae. *ICLR*, 2019. doi: 10.48550/arXiv.1809.11142.
- 624 Sheshera Mysore, Zhuoran Lu, Mengting Wan, Longqi Yang, Bahareh Sarrafzadeh, Steve Menezes,
 625 Tina Baghaee, Emmanuel Barajas Gonzalez, Jennifer Neville, and Tara Safavi. Pearl: Person-
 626 alizing large language model writing assistants with generation-calibrated retrievers. In Sachin
 627 Kumar, Vidhisha Balachandran, Chan Young Park, Weijia Shi, Shirley Anugrah Hayati, Yulia
 628 Tsvetkov, Noah Smith, Hannaneh Hajishirzi, Dongyeop Kang, and David Jurgens (eds.), *Proceed-
 629 ings of the 1st Workshop on Customizable NLP: Progress and Challenges in Customizing NLP
 630 for a Domain, Application, Group, or Individual (CustomNLP4U)*, pp. 198–219, Miami, Florida,
 631 USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.
 632 customnlp4u-1.16. URL <https://aclanthology.org/2024.customnlp4u-1.16/>.
- 633 OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Floren-
 634 cia Leoni Aleman, Diogo Almeida, Janko Altmenschmidt, Sam Altman, Shyamal Anadkat, Red
 635 Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Moham-
 636 mad Bavarian, Jeff Belgum, Irwan Bello, Jake Berdine, Gabriel Bernadett-Shapiro, Christopher
 637 Berner, Lenny Bogdonoff, Oleg Boiko, Madelaine Boyd, Anna-Luisa Brakman, Greg Brock-
 638 man, Tim Brooks, Miles Brundage, Kevin Button, Trevor Cai, Rosie Campbell, Andrew Cann,
 639 Brittany Carey, Chelsea Carlson, Rory Carmichael, Brooke Chan, Che Chang, Fotis Chantzis,
 640 Derek Chen, Sully Chen, Ruby Chen, Jason Chen, Mark Chen, Ben Chess, Chester Cho, Casey
 641 Chu, Hyung Won Chung, Dave Cummings, Jeremiah Currier, Yunxing Dai, Cory Decareaux,
 642 Thomas Degry, Noah Deutsch, Damien Deville, Arka Dhar, David Dohan, Steve Dowling, Sheila
 643 Dunning, Adrien Ecoffet, Atty Eleti, Tyna Eloundou, David Farhi, Liam Fedus, Niko Felix,
 644 Simón Posada Fishman, Juston Forte, Isabella Fulford, Leo Gao, Elie Georges, Christian Gib-
 645 son, Vik Goel, Tarun Gogineni, Gabriel Goh, Rapha Gontijo-Lopes, Jonathan Gordon, Morgan
 646 Grafstein, Scott Gray, Ryan Greene, Joshua Gross, Shixiang Shane Gu, Yufei Guo, Chris Hal-
 647 lacy, Jesse Han, Jeff Harris, Yuchen He, Mike Heaton, Johannes Heidecke, Chris Hesse, Alan
 Hickey, Wade Hickey, Peter Hoeschele, Brandon Houghton, Kenny Hsu, Shengli Hu, Xin Hu,
 Joost Huizinga, Shantanu Jain, Shawn Jain, Joanne Jang, Angela Jiang, Roger Jiang, Haozhun

- 648 Jin, Denny Jin, Shino Jomoto, Billie Jonn, Heewoo Jun, Tomer Kaftan, Łukasz Kaiser, Ali Ka-
649 mali, Ingmar Kanitscheider, Nitish Shirish Keskar, Tabarak Khan, Logan Kilpatrick, Jong Wook
650 Kim, Christina Kim, Yongjik Kim, Jan Hendrik Kirchner, Jamie Kiros, Matt Knight, Daniel
651 Kokotajlo, Łukasz Kondraciuk, Andrew Kondrich, Aris Konstantinidis, Kyle Kosic, Gretchen
652 Krueger, Vishal Kuo, Michael Lampe, Ikai Lan, Teddy Lee, Jan Leike, Jade Leung, Daniel
653 Levy, Chak Ming Li, Rachel Lim, Molly Lin, Stephanie Lin, Mateusz Litwin, Theresa Lopez,
654 Ryan Lowe, Patricia Lue, Anna Makanju, Kim Malfacini, Sam Manning, Todor Markov, Yaniv
655 Markovski, Bianca Martin, Katie Mayer, Andrew Mayne, Bob McGrew, Scott Mayer McKinney,
656 Christine McLeavey, Paul McMillan, Jake McNeil, David Medina, Aalok Mehta, Jacob Menick,
657 Luke Metz, Andrey Mishchenko, Pamela Mishkin, Vinnie Monaco, Evan Morikawa, Daniel
658 Mossing, Tong Mu, Mira Murati, Oleg Murk, David Mély, Ashvin Nair, Reiichiro Nakano, Ra-
659 jeev Nayak, Arvind Neelakantan, Richard Ngo, Hyeonwoo Noh, Long Ouyang, Cullen O’Keefe,
660 Jakub Pachocki, Alex Paino, Joe Palermo, Ashley Pantuliano, Giambattista Parascandolo, Joel
661 Parish, Emy Parparita, Alex Passos, Mikhail Pavlov, Andrew Peng, Adam Perelman, Filipe
662 de Avila Belbute Peres, Michael Petrov, Henrique Ponde de Oliveira Pinto, Michael, Pokorny,
663 Michelle Pokrass, Vitchyr H. Pong, Tolly Powell, Alethea Power, Boris Power, Elizabeth Proehl,
664 Raul Puri, Alec Radford, Jack Rae, Aditya Ramesh, Cameron Raymond, Francis Real, Kendra
665 Rimbach, Carl Ross, Bob Rotsted, Henri Roussez, Nick Ryder, Mario Saltarelli, Ted Sanders,
666 Shibanu Santurkar, Girish Sastry, Heather Schmidt, David Schnurr, John Schulman, Daniel Sel-
667 sam, Kyla Sheppard, Toki Sherbakov, Jessica Shieh, Sarah Shoker, Pranav Shyam, Szymon Sidor,
668 Eric Sigler, Maddie Simens, Jordan Sitkin, Katarina Slama, Ian Sohl, Benjamin Sokolowsky,
669 Yang Song, Natalie Staudacher, Felipe Petroski Such, Natalie Summers, Ilya Sutskever, Jie Tang,
670 Nikolas Tezak, Madeleine B. Thompson, Phil Tillet, Amin Tootoonchian, Elizabeth Tseng, Pre-
671 ston Tuggle, Nick Turley, Jerry Tworek, Juan Felipe Cerón Uribe, Andrea Vallone, Arun Vi-
672 jayvergiya, Chelsea Voss, Carroll Wainwright, Justin Jay Wang, Alvin Wang, Ben Wang, Jonathan
673 Ward, Jason Wei, CJ Weinmann, Akila Welihinda, Peter Welinder, Jiayi Weng, Lilian Weng,
674 Matt Wiethoff, Dave Willner, Clemens Winter, Samuel Wolrich, Hannah Wong, Lauren Work-
675 man, Sherwin Wu, Jeff Wu, Michael Wu, Kai Xiao, Tao Xu, Sarah Yoo, Kevin Yu, Qiming
676 Yuan, Wojciech Zaremba, Rowan Zellers, Chong Zhang, Marvin Zhang, Shengjia Zhao, Tianhao
677 Zheng, Juntang Zhuang, William Zhuk, and Barret Zoph. Gpt-4 technical report, 2024. URL
678 <https://arxiv.org/abs/2303.08774>.
- 679 Zhuoshi Pan, Qianhui Wu, Huiqiang Jiang, Xufang Luo, Hao Cheng, Dongsheng Li, Yuqing Yang,
680 Chin-Yew Lin, H. Vicky Zhao, Lili Qiu, and Jianfeng Gao. Secom: On memory construction and
681 retrieval for personalized conversational agents. In *The Thirteenth International Conference on
682 Learning Representations (ICLR)*, 2025. URL [https://openreview.net/forum?id=
683 xKDZAW0He3](https://openreview.net/forum?id=xKDZAW0He3).
- 684 Meghana Arakkal Rajeev, Rajkumar Ramamurthy, Prapti Trivedi, Vikas Yadav, Oluwanifemi Bamg-
685 bose, Sathwik Tejaswi Madhusudhan, James Zou, and Nazneen Rajani. Cats confuse reasoning
686 LLM: Query agnostic adversarial triggers for reasoning models. In *Second Conference on Lan-
687 guage Modeling*, 2025. URL <https://openreview.net/forum?id=VrEPiN5WhM>.
- 688 Shauli Ravfogel, Yanai Elazar, Hila Gonen, Michael Twiton, and Yoav Goldberg. Null it out: Guard-
689 ing protected attributes by iterative nullspace projection. In Dan Jurafsky, Joyce Chai, Natalie
690 Schluter, and Joel Tetreault (eds.), *Proceedings of the 58th Annual Meeting of the Association
691 for Computational Linguistics*, pp. 7237–7256, Online, July 2020. Association for Computational
692 Linguistics. doi: 10.18653/v1/2020.acl-main.647. URL [https://aclanthology.org/
693 2020.acl-main.647/](https://aclanthology.org/2020.acl-main.647/).
- 694 Chris Richardson, Yao Zhang, Kellen Gillespie, Sudipta Kar, Arshdeep Singh, Zeynab Raeesy,
695 Omar Zia Khan, and Abhinav Sethy. Integrating summarization and retrieval for enhanced per-
696 sonalization via large language models. 2023.
- 697 Alireza Salemi, Sheshera Mysore, Michael Bendersky, and Hamed Zamani. LaMP: When large
698 language models meet personalization, 2023.
- 699 Alireza Salemi, Surya Kallumadi, and Hamed Zamani. Optimization methods for personaliz-
700 ing large language models through retrieval augmentation. SIGIR ’24, pp. 752–762, New
701 York, NY, USA, 2024. Association for Computing Machinery. ISBN 9798400704314. doi:
10.1145/3626772.3657783. URL <https://doi.org/10.1145/3626772.3657783>.

- 702 Valter Schütz, Han Wu, Reza Rezvan, Linus Aronsson, and Morteza Haghir Chehreghani. Afabench:
703 A generic framework for benchmarking active feature acquisition, 2025. URL [https://](https://arxiv.org/abs/2508.14734)
704 arxiv.org/abs/2508.14734.
705
706
707
708
709 Taylor Sorensen, Jared Moore, Jillian Fisher, Mitchell Gordon, Niloofar Mireshghallah, Christo-
710 pher Michael Rytting, Andre Ye, Liwei Jiang, Ximing Lu, Nouha Dziri, Tim Althoff, and Yejin
711 Choi. Position: a roadmap to pluralistic alignment. In *Proceedings of the 41st International*
712 *Conference on Machine Learning*, ICML'24. JMLR.org, 2024.
713
714
715
716 Zhaoxuan Tan, Zheyuan Liu, and Meng Jiang. Personalized pieces: Efficient personalized large
717 language models through collaborative efforts. In *Proceedings of the 2024 Conference on Empir-*
718 *ical Methods in Natural Language Processing (EMNLP)*, pp. 6459–6475. Association for Com-
719 putational Linguistics, 2024. URL [https://aclanthology.org/2024.emnlp-main.](https://aclanthology.org/2024.emnlp-main.371.pdf)
720 [371.pdf](https://aclanthology.org/2024.emnlp-main.371.pdf).
721
722
723
724 Meiling Tao, Chenghao Zhu, Dongyi Ding, Tiannan Wang, Yuchen Eleanor Jiang, and Wangchun-
725 shu Zhou. Personafeedback: A large-scale human-annotated benchmark for personalization,
726 2025. URL <https://arxiv.org/abs/2506.12915>.
727
728
729
730
731 Jörg Tiedemann and Santhosh Thottingal. OPUS-MT — Building open translation services for the
732 World. In *Proceedings of the 22nd Annual Conferenec of the European Association for Machine*
733 *Translation (EAMT)*, Lisbon, Portugal, 2020.
734
735
736
737 Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. Minilm: Deep
738 self-attention distillation for task-agnostic compression of pre-trained transformers, 2020. URL
739 <https://arxiv.org/abs/2002.10957>.
740
741
742
743 Haoxing Zhang, Xiaofeng Zhang, Haibo Huang, and Lei Yu. Prompt-based meta-learning for
744 few-shot text classification. In Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang (eds.), *Pro-*
745 *ceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pp.
746 1342–1357, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational
747 Linguistics. doi: 10.18653/v1/2022.emnlp-main.87. URL [https://aclanthology.org/](https://aclanthology.org/2022.emnlp-main.87/)
748 [2022.emnlp-main.87/](https://aclanthology.org/2022.emnlp-main.87/).
749
750
751
752
753 Wenting Zhao, Xiang Ren, Jack Hessel, Claire Cardie, Yejin Choi, and Yuntian Deng. Wildchat:
754 1m chatGPT interaction logs in the wild. In *The Twelfth International Conference on Learning*
755 *Representations (ICLR 2024)*, *Spotlight Poster*, 2024. URL [https://openreview.net/](https://openreview.net/forum?id=B18u7ZR1bM)
[forum?id=B18u7ZR1bM](https://openreview.net/forum?id=B18u7ZR1bM).

A APPENDIX

B ADDITIONAL EXPERIMENTATION

In this section, we present additional experiments to supplement our main results.

B.1 ROBUSTNESS ANALYSIS OF UTILITY SCORES

In H1, we hypothesized that RUMS-Utility, the maximum utility scores, can reliably distinguish between user inputs that would benefit from personalization and those that would not. Here, we further investigate how hyperparameters in the RUMS-Utility algorithm affect this reliability and provide justification for these effects. Specifically, we examine the influence of model type, model size, decoding method type, maximum number of tokens generated, number of Monte Carlo samples, and number of user profiles. For each analysis, all other variables are held constant using the LLaMa base model, 8B size, sampling decoding, $T = 20$ tokens, $n = 1$ sample, and averaged over $n = 3$ user profiles.

Hyperparameter: Model Type We first compare the LLaMa-8B base model to an LLaMa-8B instruction-tuned model. As shown in Figure 4, the base model produces more separated distributions of utility scores than the instruct model, with a Jensen-Shannon divergence score of 0.56 compared to 0.32 for the instruct model. This is consistent with prior research showing that instruction-tuning tends to pre-sharpen output distributions (Sorensen et al., 2024). Based on this observation, we chose to use the base model for our experiments.

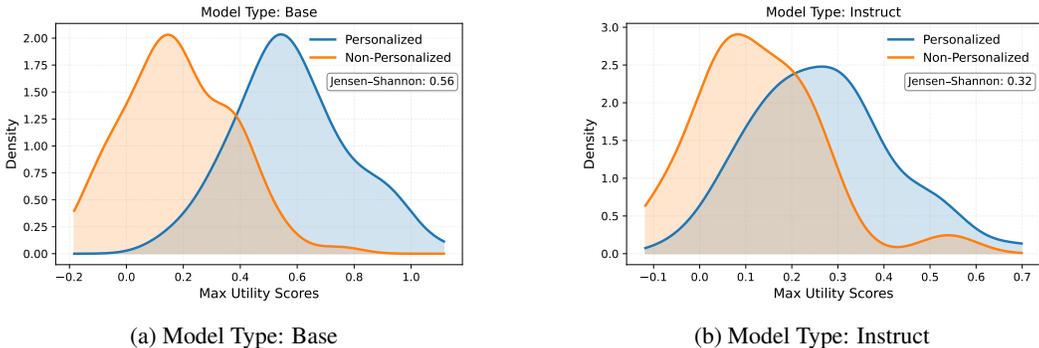


Figure 4: Distribution of maximum utility scores for a personalized (blue) and non-personalized (orange) dataset differing by type of model used for entropy calculation.

Hyperparameter: Model Size Next, we examined the effect of model size on the entropy calculation. As shown in Figure 5, the distribution of utility scores changes when using LLaMA-1B compared to LLaMA-8B. While the impact is less pronounced than the effect of model type, we observe an increase of 0.11 in the Jensen-Shannon divergence score when using the 8B model. We hypothesize that larger models are better able to capture subtle and complex patterns in their response distributions. Given that the improvement is relatively modest, we opted to use the 8B model to balance performance with computational efficiency and avoid the added cost of larger models.

Hyperparameter: Decoding Type We next examine the effect of decoding type on the sequential average token entropy, comparing greedy and sampling decoding. As shown in Figure 6, sampling decoding produces more diverse distributions than greedy decoding. This is expected, as sampling introduces stochasticity into token selection, allowing the model to explore multiple plausible continuations, whereas greedy decoding always selects the highest-probability token at each step, leading to more concentrated and less variable distributions. Based on this observation, we use sampling decoding in our experiments.

810
811
812
813
814
815
816
817
818
819
820
821

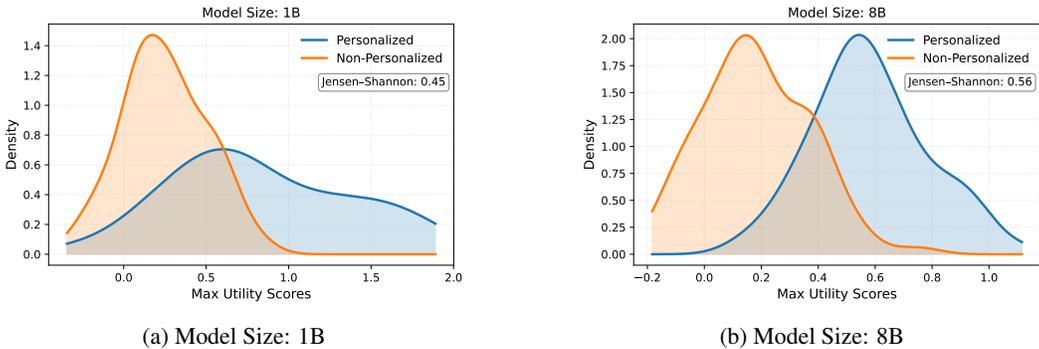


Figure 5: Distribution of maximum utility scores for a personalized (blue) and non-personalized (orange) dataset differing by size of model used for entropy calculation.

822
823
824
825
826
827
828
829
830
831
832
833
834
835
836

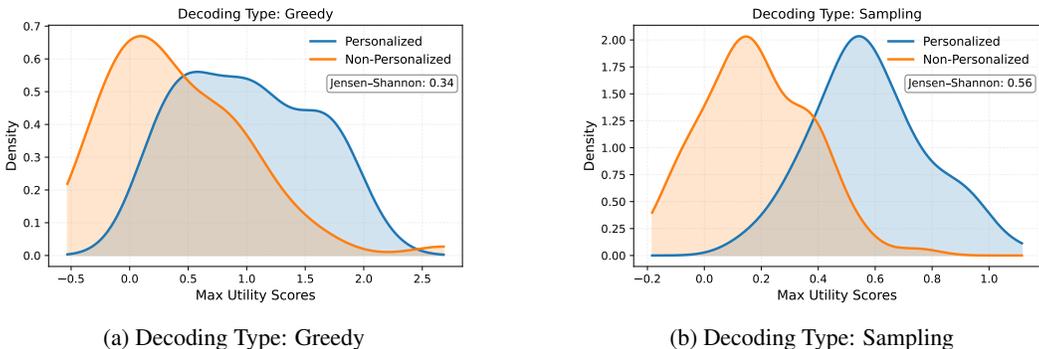


Figure 6: Distribution of maximum utility scores for a personalized (blue) and non-personalized (orange) dataset differing by decoding type.

840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863

Hyperparameter: Number of Tokens As noted in the main text, we limit the number of tokens used in the calculation of Equation (5) to avoid bias introduced by longer generations, since longer outputs tend to have lower entropy. To study this, we randomly sampled $n = 10,000$ responses from the OpenAssistant dataset (Köpf et al., 2023), using only the first human input in each conversation that was labeled as English and non-toxic. We used this dataset since it is a real world dataset that could provide a wide diversity in types of user-LLM interactions. We then computed the average token-level entropy by token index, capped at 100 tokens. As shown in Figure 7, entropy begins to decrease significantly around $T = 20$ tokens, indicating that longer generations introduce a bias toward lower entropy.

Further analysis in Figure 8 compares the distribution of maximum utility scores when using 20 versus 100 maximum tokens for entropy calculations. Using 100 tokens reduces the Jensen-Shannon divergence by 0.18 compared to using only 20 tokens. Based on these results, we limit the calculation to the first 20 tokens in our experiments.

Hyperparameter: Number of Samples Next, we examine the effect of the number of Monte Carlo samples used in Equation (5) on the robustness of the utility score. As shown in Figure 9, increasing the number of samples improves the separation between personalized and non-personalized inputs, which aligns with expectations since the estimate should converge toward the true parameter as the sample size increases if the samples are i.i.d. However, this comes with a tradeoff: more samples require additional computation, longer runtime, and higher resource costs. Balancing these considerations, we chose $n = 5$ samples for our experiments.

Hyperparameter: Number of User Profiles Lastly, we examine how the robustness of the average maximum utility score changes as we increase the number of (query,user profile) pairs included in the average. Consistent with estimation theory, we find that the difference between the person-

864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885

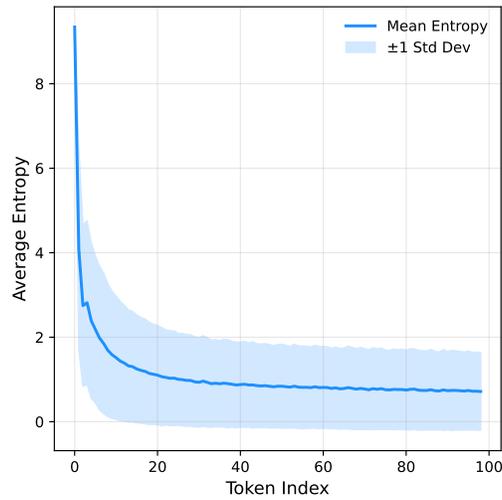
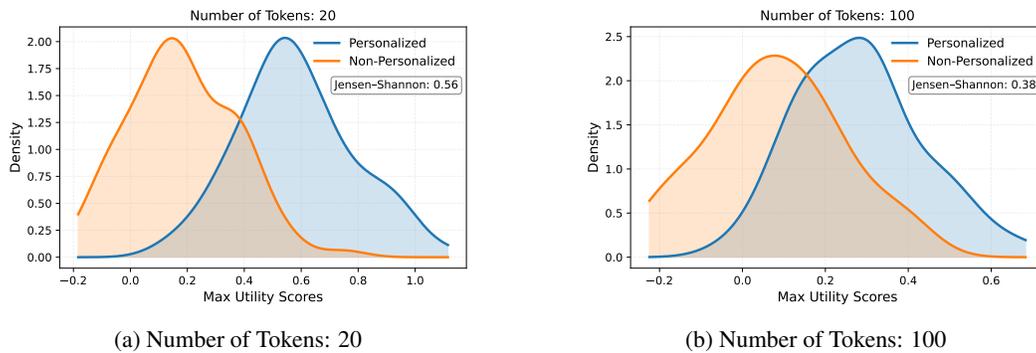
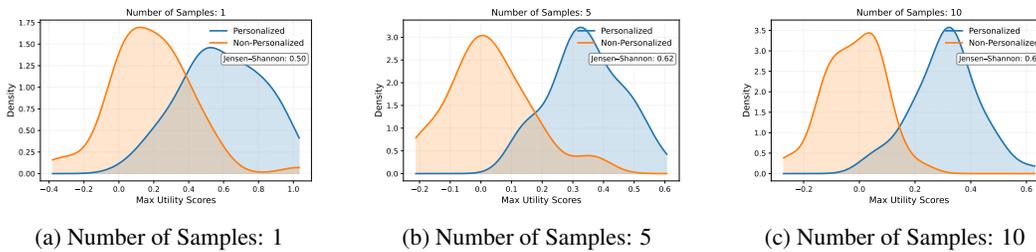


Figure 7: Average entropy by the index of the token being generated.



886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907



908
909
910
911
912
913
914
915
916
917

alized and non-personalized distributions grows as the sample size increases. Notably, there is a substantial jump of 0.22 in Jensen–Shannon divergence when increasing from 1 to 5 users, but only a marginal increase of 0.01 when going from 5 to 10 users. Based on this, we choose $u = 5$ users for constructing our training data to balance performance with cost of inference.

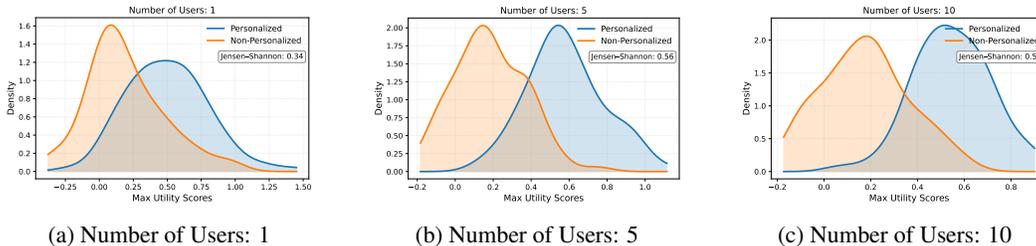


Figure 10: Distribution of maximum utility scores for a personalized (blue) and non-personalized (orange) dataset differing by number of users averaged over.

B.2 RUMS-UTILITY THRESHOLD ANALYSIS

As mentioned in the main text, RUMS-Utility has a threshold hyperparameter that controls the sensitivity of memory item selection. Adjusting this threshold results in either more or fewer memory items being chosen. Table 4 illustrates how the average number of selected memory items changes as the threshold is varied. As the threshold increases, a stronger reduction in entropy is required for a memory item to be selected, leading to smaller average subset sizes. This analysis highlights that the threshold provides a simple yet effective way to balance the tradeoff between including more context and avoiding unnecessary or low-utility memory items.

Threshold	PersonaFeedback	FreebaseQA	WildChat-Personal	WildChat-NonPersonal
-100	2.76	1.83	2.77	2.70
0.00	2.76	1.56	2.73	2.57
0.29	2.43	0.21	1.31	1.38
0.50	1.11	0.00	0.29	0.27
0.80	0.14	0.00	0.00	0.00

Table 4: Average number of memory items selected at different thresholds used in RUMS-Utility.

B.3 EXAMPLES OF SELECTED MEMORY ITEMS

Table 5 illustrates one personalized and one non-personalized case. Consistent with Table 2, GPT4-ZeroShot and Semantic Similarity over-select, adding noise in non-personalized cases; GPT4-FewShot selects less in general but still selects items for non-personalized inputs. RUMS-Binary balances personalization by matching human annotations in the personalized case and abstaining in the non-personalized case.

B.4 COST ANALYSIS

To better understand the practical implications of our approach, we analyze the tradeoff between computational cost and model performance across all methods. Since our methods and baselines differ in the number of memory items they select, and hence the number of additional tokens they introduce into the prompt, it is important to quantify how these differences affect overall efficiency at scale.

We measure computation overhead with two complementary metrics. First, we calculate the *average number of memory items selected* per dataset and method. This captures how the amount of items each method chooses in order to fulfill personalization or how little for no personalization. Second, we compute the *average number of additional tokens* that must be appended to the prompt during inference in order to include the selected memory items. This provides a direct measure of the inference overhead. Table 6 and Table 7 summarize these results, showing clear differences across methods: baselines such as All or Random predictably add large numbers of items, while information-retrieval heuristics like BM25 are highly selective. Our methods lie in between, striking a balance between selecting informative items and avoiding unnecessary additions.

Table 5: Examples of memory items selected by each method compared to human gold labels. Overlapping memory with items human annotation is **bolded**.

User Input	Humans*	RUMS-Binary	GPT4-ZeroShot	GPT4-FewShot	Semantic Similarity
[PERSONALIZED] I became a millionaire recently and want to give money to some local businesses	Location Hobbies and Interests Volunteer Activities	Location Hobbies and interests Volunteer activities Personal values Industry of employment	Location Occupation Industry of employment Current projects Long-term aspirations	Location Recent life events	Location Volunteer activities Financial situation Annual Income Pet ownership
[NON-PERSONALIZED] Who was the Prime Minister of Gt. Britain at the outbreak of The First World War?	No features selected	No features selected	Location Occupation Name Political affiliation	Location Occupation	Travel history Occupation Name Political affiliation Current projects

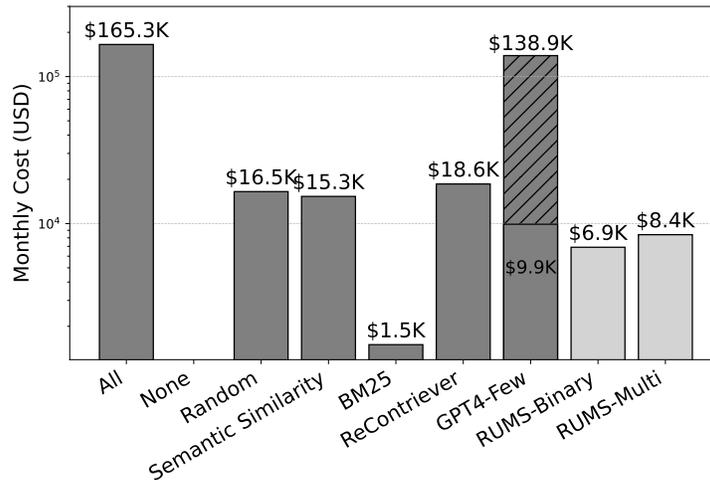


Figure 11: Estimated monthly cost of additional prompt tokens (1M queries per day, 30 days, GPT-4 input pricing at \$0.01 per 1K tokens).

To translate these measures into practical cost, we estimate the monthly cost of running one million queries per day for each method using pricing aligned with GPT-4. Specifically, we convert the additional tokens from Table 2 into dollar cost under the assumption that each token incurs a fixed marginal cost of \$0.01 per 1K tokens and 30 days per month. This allows us to approximate the total monthly inference cost for each method, and thus assess the economic tradeoff of personalization strategies at scale.

The results are visualized in the Figure 11. As expected, the All baseline is by far the most expensive, exceeding \$160K per month due to its maximal inclusion of all memory items. On the other hand, our model RUMS-Binary yields much lower additional cost of only \$6.9K which is about 30% lower than just the inference cost of including the memory features selected by GPT4-FewShot, not including the cost to generate these memory items. Our other model, RUMS-Multi, also has low cost of only \$8.4K, keeping costs well below stronger baselines like GPT-4, ReContriever, and Semantic Similarity. Overall, this analysis highlights that RUMS-Models can be deployed at scale in a cost-efficient manner, with our methods demonstrating favorable tradeoffs between informativeness and economic feasibility.

B.5 FULL RESULTS OF H3

In this section we expand H3 by breaking down results for the Synthetic and Real World datasets by inputs intended to be personalized (left) and not personalized (right). This comparison highlights differences between RUMS-Binary and RUMS-Multi. With the less capable response model, LLaMA-70B, RUMS-Binary (which favors fewer memory items) outperforms most baselines, showing es-

Table 6: Average number of memory items selected by method per dataset.

Method	FreebaseQA	PersonaFeedback	WildChat-Personal	WildChat-NonPersonal
All	50	50	50	50
None	0	0	0	0
Random	5	5	5	5
Semantic Similarity	4.71	4.54	4.84	4.74
BM25	0.39	0.81	0.28	0.42
ReContriever	5	5	5	5
GPT4-Few	1.21	4.39	3.90	2.10
RUMS-Binary	0.02	3.14	2.80	1.64
RUMS-Multi	1.45	3.58	2.36	1.76

Table 7: Average number of extra tokens required for response generation.

Method	FreebaseQA	PersonaFeedback	WildChat-Personal	WildChat-NonPersonal
All	551.59	551.28	552.16	550.40
None	0	0	0	0
Random	54.87	54.98	54.34	54.20
Semantic Similarity	48.04	51.44	53.04	49.58
BM25	1.48	8.95	3.20	4.86
ReContriever	60.87	62.93	61.22	61.72
GPT4-Few	12.96	49.31	46.02	23.58
RUMS-Binary	0.23	37.39	34.88	19.10
RUMS-Multi	13.88	45.34	30.90	21.08

pecially strong gains on non-personalized inputs, with a 38% average win rate over baselines on the Synthetic dataset. Similar patterns hold on the Real World dataset, except for the None and BM25 baselines, which never select items.

By contrast, RUMS-Multi (which selects more memory items) performs better with the stronger GPT-4 model, particularly on personalized inputs, achieving 25% and 26% average win rates on the personalized Synthetic and Real World datasets respectively. These results demonstrate how RUMS can be tuned to different settings, adapting to the response model and personalization needs.

Table 8: Average winrate comparison on Synthetic data under Not Personal (left) and Personal (right) conditions. Positive values (green) indicate better performance, negative values (red) indicate worse performance. Values greater than 0.1 in magnitude are bolded.

	(a) Non-personalized inputs				(b) Personalized inputs				
	LLaMA 70B		GPT-4		LLaMA 70B		GPT-4		
	RUMS-Binary.	RUMS-Multi	RUMS-Binary.	RUMS-Multi	RUMS-Binary.	RUMS-Multi	RUMS-Binary.	RUMS-Multi	
None	-0.03	-0.79	0.02	0.00	none	0.52	0.53	0.50	0.72
All	0.45	-0.24	0.00	0.02	All	-0.14	-0.14	-0.31	0.01
Random	0.50	-0.12	-0.03	-0.01	Random	0.18	0.37	0.15	0.39
Semantic Similarity	0.60	0.01	0.02	0.00	Semantic Similarity	-0.06	0.17	-0.07	0.09
BM25	0.12	-0.66	-0.02	0.01	BM25	0.25	0.40	0.42	0.40
ReContriever	0.62	-0.14	0.01	0.03	ReContriever	-0.09	0.09	-0.09	0.09
GPT4-Few	0.42	-0.34	0.00	0.01	GPT4-Few	-0.07	0.03	-0.17	0.03

C ADDITIONAL RELATED WORKS

Foundations and Variations of BOED One of the earliest Bayesian methods for experimental design was proposed by Lindley (1956), which introduced a general framework based on maximizing the expected Shannon entropy. Bernardo (1979) later extended this work to incorporate decision spaces. Since then, BOED has been explored in numerous settings; for a complete review, see Chaloner & Verdinelli (1995). However, in practice, BOED can be computationally demanding because it requires estimating the expected information gain of each candidate experiment. To address

Table 9: Average winrate comparison on Real World data under Not Personal (left) and Personal (right) conditions. Positive values (green) indicate better performance, negative values (red) indicate worse performance. Values greater than 0.1 in magnitude are bolded.

	(a) Non-personalized inputs				(b) Personalized inputs			
	LLaMA 70B		GPT-4		LLaMA 70B		GPT-4	
	RUMS-Binary	RUMS-Multi	RUMS-Binary	RUMS-Multi	RUMS-Binary	RUMS-Multi	RUMS-Binary	RUMS-Multi
None	-0.36	-0.64	-0.12	-0.26	0.46	0.70	0.52	0.60
All	0.08	-0.14	-0.14	-0.16	-0.16	-0.04	-0.14	0.04
Random	-0.02	-0.06	-0.06	-0.26	0.24	0.42	0.26	0.34
Semantic Similarity	0.04	-0.02	-0.02	-0.12	-0.10	0.00	-0.02	0.12
BM25	-0.40	-0.16	-0.16	-0.26	0.42	0.58	0.52	0.52
ReContriever	0.12	0.04	0.04	0.02	0.00	0.24	-0.14	0.24
GPT4-Few	0.08	0.12	0.12	-0.02	-0.08	-0.04	-0.04	-0.04

this challenge, Foster et al. (2019) introduced a variational BOED approach that leverages amortized inference to efficiently approximate the expected information gain for experiment selection. Similarly, Schütz et al. (2025) formalized adaptive submodularity, showing that greedy information-maximization strategies, such as selecting the feature with the highest expected entropy reduction at each step, are near-optimal in many cases. These works provide the foundational justification for our estimations used in RUMS.

D FORMULAS

D.1 SHANNON ENTROPY

We define the Shannon Entropy as a measure of uncertainty within a system. For a random variable $x \in X$ and a distribution according to $p : X \rightarrow [0, 1]$ such that $p(x) := \mathbb{P}[X = x]$. Then we formalize the Shannon entropy H as,

$$H(x) = - \sum_{x \in X} p(x) \log p(x)$$

E SEQUENCE ENTROPY EQUIVALENCE PROOF

Proof. Let $p_\theta(y | x)$ denote the model distribution over token sequences $y = (y_1, \dots, y_T)$ given input x . At each step t , the conditional next-token distribution is

$$p_\theta(\cdot | y_{<t}, x),$$

where $y_{<t}$ is the prefix of tokens before step t .

By the chain rule of probability,

$$-\log p_\theta(y | x) = \sum_{t=1}^T -\log p_\theta(y_t | y_{<t}, x).$$

Taking the expectation with respect to $y \sim p_\theta(\cdot | x)$ on both sides gives

$$\begin{aligned} \mathbb{E}_{y \sim p_\theta} [-\log p_\theta(y | x)] &= \mathbb{E}_{y \sim p_\theta} \left[\sum_{t=1}^T -\log p_\theta(y_t | y_{<t}, x) \right] \\ &= \sum_{t=1}^T \mathbb{E}_{y \sim p_\theta} [-\log p_\theta(y_t | y_{<t}, x)] && \text{linearity of expectation} \\ &= \sum_{t=1}^T \mathbb{E}_{y_{<t} \sim p_\theta} \left[\sum_{y_t} p_\theta(y_t | y_{<t}, x) - \log p_\theta(y_t | y_{<t}, x) \right] && \text{law of iterated expectation} \\ &= \mathbb{E}_{y_{<t} \sim p_\theta} \left[\sum_{t=1}^T H(p_\theta(\cdot | y_{<t}, x)) \right]. \end{aligned}$$

□

F RUMS-UTILITY ALGORITHM

In Algorithm 1 we provide the formalized algorithm for RUMS-Utility.

Algorithm 1 Memory Selection via RUMS

Require: User input x , user memory $u \in \mathbb{R}^d$
Ensure: Selected memory subset $m_{u,S}^*$ that maximizes utility

- 1: **Candidate Memory Item Identification:**
- 2: Use LLM to generate candidate subsets S for x
- 3: **Subset Evaluation:**
- 4: **for** each $s \in S$ **do**
- 5: Map candidate attributes to user values $m_{u,S}$
- 6: Prompt Reference LM with:
- 7: (i) x only
- 8: (ii) x and $m_{u,S}$
- 9: Compute estimated utility $\text{EIG}(S)$ using Equation (5)
- 10: Store utility score for subset s
- 11: **Subset Selection:**
- 12: $s^* \leftarrow \arg \max_{s \in S} \text{EIG}(S)$
- 13: **if** all MI scores $<$ threshold **then**
- 14: $s^* \leftarrow \emptyset$ ▷ Select no features

G EXPERIMENTAL DETAILS

This section provides the full details of the experiments described in Section 4.

G.1 EXPERIMENTAL SETUP

Across all experiments (H1–H3), we consider a fixed set of $n = 50$ memory items (e.g., *location*, *age*, *dietary restrictions*). The task is to select a subset of up to $n = 5$ memory items that can improve personalization.

G.2 DATASETS

To evaluate scenarios where personalization is beneficial as well as those where it is not, we use a diverse mixture of user inputs drawn from prior work, including both synthetic and real-world sources. The three datasets we used are described below:

PersonaFeedback (Tao et al., 2025) This is a synthetically created dataset consisting of user-specific queries designed to reflect inputs from realistic and diverse user personas. First, they constructed personas by representing them as a structured dictionary of memory items, including demographics, personality traits, and preferences. Then, open-source content from social media, reviews, and forums were used to populate these items with specific values. Lastly, an LLM generator was used to produce queries based on a sampled persona and specific sub-scenarios. Additional steps such as similarity-based filtering, rephrasing to avoid directly revealing persona information, and manual annotations ensured diversity and realism. The final dataset contains more than 4,000 persona-specific questions. The original data is in Chinese and therefore, we translated to English using a machine translation model (Tiedemann & Thottingal, 2020). Example: “Do you have any music for the trip?”

FreebaseQA (Jiang et al., 2019) A dataset of trivia-style questions and answers collected from human trivia competitions. The original dataset contains approximately 28,000 question–answer pairs. Example: “What type of snake is a sidewinder?”

1188 **WildChat (Zhao et al., 2024)** A large-scale collection of more than 1 million real-world
 1189 user–ChatGPT interactions, encompassing over 2.5 million dialogue turns. The dataset was com-
 1190 piled by offering free access to ChatGPT and GPT-4 in exchange for user consent to share chat
 1191 histories. It covers diverse conversational scenarios such as ambiguous requests, code-switching,
 1192 topic shifts, and political discussions. We filter the data to retain only English conversations labeled
 1193 as non-toxic, and we use only the first human-written input per conversation to ensure that queries
 1194 contain sufficient context.

1195 To determine whether an input is personalized, two human annotators labeled a random selection
 1196 of WildChat queries as either beneficial for personalization (WildChat-Personal) or not (WildChat-
 1197 NonPersonal). Only queries with full annotator agreement were retained. Examples: “*I’m hungry*”;
 1198 “*Who is Mohamed Gamal Mohamed Maniieh?*”
 1199

1200 **Synthetic vs. Real-World Datasets** By design, PersonaFeedback is treated as a personalized
 1201 input set, and FreebaseQA as non-personalized. We collectively refer to these as the “Synthetic”
 1202 dataset, since they were not originally created for LLM input. WildChat is considered a “Real-
 1203 World” dataset, as it was collected directly from real user interactions with an LLM. We use both
 1204 synthetic and real-world datasets to evaluate the robustness of RUMS across many diverse inputs.
 1205

1206 **Sample Size by Hypothesis** The number of samples varies across hypotheses. For H1, we use
 1207 $n = 500$ samples from PersonaFeedback and FreebaseQA to analyze RUMS-Utility for binary per-
 1208 sonalization decisions. For H1’, we use $n = 25$ inputs each from PersonaFeedback, FreebaseQA,
 1209 WildChat-Personal, and WildChat-NonPersonal to extend the H1 analysis to correlation with human
 1210 judgment. For H2, we use $n = 15$ inputs each from PersonaFeedback, FreebaseQA, WildChat-
 1211 Personal, and WildChat-NonPersonal to examine the correlation between RUMS memory selection
 1212 and human preference. Finally, for H3, we use $n = 100$ samples from PersonaFeedback and Free-
 1213 baseQA along with $n = 50$ samples each from WildChat-Personal and WildChat-NonPersonal to
 1214 analyze the downstream impact of subsets chosen by RUMS-Model.

1215 G.3 PROFILE GENERATION

1216 User profiles are incorporated during the training data curation phase, where specific user values are
 1217 used to compute the entropy difference of the model required for the utility function in Equation (5).
 1218 Also, we use profiles in our test sets for evaluation.
 1219

1220 **Memory Item Creation** To construct user profiles, we prompted GPT-4 (OpenAI et al., 2024) to
 1221 generate 50 commonly occurring user memory items. These items span demographic, behavioral,
 1222 and preference-based attributes, providing a diverse set of values for personalization. The complete
 1223 list is as follows:
 1224

- 1225 • Name
- 1226 • Gender
- 1227 • Location (City/State/Country)
- 1228 • Ethnicity
- 1229 • Race
- 1230 • Annual Household Income
- 1231 • Language spoken at home
- 1232 • Language preference
- 1233 • Sexual Preference
- 1234 • Disabilities
- 1235 • Occupation
- 1236 • Political affiliation
- 1237 • Education level
- 1238 • Hobbies and interests
- 1239 • Favorite sports
- 1240 • Preferred music genre
- 1241 • Favorite books
- Preferred movie genre
- Favorite TV shows
- Favorite foods
- Dietary restrictions (e.g.,
vegetarian, vegan)
- Relationship status
- Number of children
- Pet ownership (types of
pets)
- Travel history (countries
visited)
- Travel preferences (adven-
turous, relaxing)
- Social media platforms used
- Level of tech-savviness
- Preferred mode of commu-
nication (text, voice, video)
- Work schedule (full-time,
part-time)
- Industry of employment
- Current projects or goals
- Long-term aspirations
- Financial situation (broad
categories)
- Health and fitness goals
- Preferred exercise activities

- 1242 • Level of environmental con- • Favorite pastimes • Frequency of using the ser-
- 1243 sciousness • Personal values or beliefs vice (daily, weekly)
- 1244 • Volunteer activities or inter- • Cultural background • Reasons for using the ser-
- 1245 ests • Religious beliefs vice
- 1246 • Current challenges or pain • Recent life events (e.g., • Feedback preferences (de-
- 1247 points change in job, moved) tailed, brief)
- 1248 • Personality traits (e.g., intro- • Technology usage (devices, • Preferred tone of communi-
- 1249 verted, extroverted) apps cation (formal, casual)
- 1250
- 1251

1252 **Memory Item Values** Next, we used GPT-4 (OpenAI et al., 2024) to generate consistent user
 1253 profiles without contradictions (e.g., “vegan” with “favorite food = steak”), following the prompt in
 1254 Prompt 1. To further ensure logical consistency, we employed GPT-4 to filter out any profiles that
 1255 appeared illogical, as described in Prompt 2. This process was repeated until we obtained $n = 100$
 1256 unique, logically consistent profiles for both training and testing.

1257 This method yields a diverse set of user profiles, as illustrated in the bar charts of memory item
 1258 distributions shown in Figure 12 through Figure 61.

1259

1260 Prompts used for Profile Generation

1261 **Prompt 1.** *For these 50 features, I want you to create 10 very diverse persona profiles. So for each*
 1262 *of the 50 features, I want you to select a value (e.g. "age" = 50) that would make up the persona*
 1263 *profile. Make sure the values make sense for each other, meaning don't have a person who is 12*
 1264 *years old have a job as a accountant.*

1265 *Output a json file with a dictionary with 10 keys, "Persona_1", "Persona_2", ..., "Persona_10", whose*
 1266 *corresponding values is a list of the selected values in the same order as the features (e.g. [John,*
 1267 *Male, Seattle/WA/USA, ...]).*

1268 *Do not include any other markdown or text. Here are the Features:*

1269 **Prompt 2.** *You are an AI assistant that needs to decide if a user profile is logically sound. This*
 1270 *means that the information in the user profile should be consistent and non-conflicting.*

1271 *Only output a single word, either 'yes' or 'no', to indicate if the user profile is logically sound.*

1272 *Example 1: If a user profile states that the user is allergic to peanuts, it should not also state their*
 1273 *favorite food is peanuts.*

1274 *Example 2: If a user profiles states highest education is a GED, it should not also state their profes-*
 1275 *sion is a doctor.*

1276 *Example 3: If a user profile states that the user is 25 years old, it should not also state that they have*
 1277 *been working for 30 years.*

1278 *Example 4: If a user profile states that they live in Columbia, it should not state that their main*
 1279 *language is French.*

1280 *User Profile:*

1281

1282 G.4 BASELINES

1283

1284 In this section we describe each baseline that is used in our experimentation.

1285

1286 **GPT-4 Prompting (OpenAI et al., 2024):** We use GPT-4o as a baseline across H1, H2, and H3.
 1287 In H1, we apply Prompt 3 for the zero-shot setting and Prompt 4 for the few-shot setting to decide
 1288 whether personalization is needed for a given query. For the few-shot case, we provide four in-
 1289 context examples: two requiring personalization and two not requiring personalization.

1290 In H2 and H3, we adapt the prompts to instead elicit the subset of specific user memory items that
 1291 would be beneficial for personalization. We use Prompt 5 for zero-shot and Prompt 6 for few-shot.
 1292 In the few-shot setting, we provide two in-context examples: one requiring personalization and one
 1293 not.

1296 **Prompt 3.** *You are a classifier that determines whether a given user input would benefit from user-*
1297 *specific information for the model to respond.*

1298 *Your response must be a JSON object with a two key "Decision" with a value of either "True" or*
1299 *"False" and "Features" with either a list of user-specific features (at most 10 features) if "Decision"*
1300 *is "True" or "None" if "Decision" is "False" :*
1301

1302 *Output "True" if the input would benefit from user-specific information to better respond to the*
1303 *query.*

1304 *Output "False" if the input would not benefit from user-specific information or would be the same*
1305 *answer regardless of the user.*

1306 *Respond with only the JSON object. Do not include any explanation or additional text.*
1307

1308 *Input: user_input*

1309 **Prompt 4.** *'You are a classifier that determines whether a given user input would benefit from*
1310 *user-specific information for the model to respond.*

1311 *Your response must be a JSON object with a two key "Decision" with a value of either "True" or*
1312 *"False" and "Features" with either a list of user-specific features (at most 10 features) if "Decision"*
1313 *is "True" or "None" if "Decision" is "False" :*
1314

1315 *Output "True" if the input would benefit from user-specific information to better respond to the*
1316 *query.*

1317 *Output "False" if the input would not benefit from user-specific information or would be the same*
1318 *answer regardless of the user.*

1319 *Respond with only the JSON object. Do not include any explanation or additional text.*
1320

1321 *Example 1:*

1322 *Input: "Where should I eat tonight?"*

1323 *Output: "Decision": "True", "Features": ["location", "dietary restrictions", "income"]*

1324 *Example 2:*

1325 *Input: "Can you help me write an essay on feminism?"*

1326 *Output: "Decision": "True", "Features": ["education", "political ideology", "gender"]*

1327 *Example 3:*

1328 *Input: "What is the capital of Austin, Texas?"*

1329 *Output: "Decision": "False", "Features": "None"*

1330 *Example 4:*

1331 *Input: "Can you explain what AGI is?"*

1332 *Output: "Decision": "False", "Features": "None"*

1333 *Now classify the following input:*

1334 *Input: user_input'*

1335 **Prompt 5.** *Given a user-input and list of features, select which features might be important for*
1336 *answering the user-input.*

1337 *Choose up to 5 features. Respond in a JSON with one key "Chosen Features", which has a list of*
1338 *chosen features. Do not include any other text or formatting.*

1339 *User Input: user_input*

1340 *Possible Features: candidate_features_ls*

1341 **Prompt 6.** *Given a user-input and list of features, select which features might be important for*
1342 *answering the user-input. Choose up to 5 features.*
1343
1344
1345
1346
1347
1348
1349

1350 *Respond in a JSON with one key "Chosen Features", which has a list of chosen features. Do not*
 1351 *include any other text or formatting.*

1352 *Example Input 1: "whats the weather today"*

1354 *Example Output 1: "Chosen Features": ["Location (City/State/Country)"]*

1355 *Example Input 2: "What is the capital of Texas?"*

1357 *Example Output 2: "Chosen Features": []*

1358 *User Input: user_input*

1360 *Possible Features: candidate_features_Is'*

1362 **Semantic Similarity:** In H1–H3, we also include a simple baseline based on semantic similarity.
 1363 Here, we compute the cosine similarity between user query embeddings and each memory item
 1364 embeddings using a lightweight sentence-transformer model (Wang et al., 2020).

1365 In H1, we use the validation set, defined as a 20% hold-out of the data, to tune a threshold for
 1366 deciding whether personalization is required. Thresholds are determined by grid search over $[-1, 1]$
 1367 in increments of 0.5, selecting the value that maximizes agreement with human annotations for
 1368 each dataset. A query is labeled as requiring personalization if at least one memory item has a
 1369 similarity score above the selected threshold. The exact thresholds used for each dataset are reported
 1370 in Table 10.

1371

Dataset	Threshold
All	-1.0
Synthetic	-1.0
Real World	0.5

1376 Table 10: Threshold for H1 analysis of semantic similarity baseline. If at least one memory item
 1377 had a semantic similarity higher than this threshold, then the query was labeled as needing person-
 1378 alization.

1379

1380 Since H2 and H3 require selecting specific memory items for personalization rather than making
 1381 a simple binary decision, we set an intuitive threshold of $t = 0$, corresponding to no similarity
 1382 between the user query and a memory item. If more than five memory items exceed this threshold,
 1383 we select the top five with the strongest correlations.

1385 **Simple Baselines:** We also compare against several naive baselines. The “None” baseline pro-
 1386 vides no features ($n = 0$), serving as a lower bound for personalized inputs and an upper bound for
 1387 non-personalized ones. The “All” baseline includes all available features ($n = 50$), testing the effect
 1388 of providing maximal but noisy context. Finally, the “Random” baseline samples $n = 5$ items at
 1389 random, establishing a chance-level reference.

1390

1391 **SOTA Retrieval Methods:** We additionally evaluate two state-of-the-art retrieval methods.

1393 First, BM25 (Lù, 2024), a widely used ranking function in information retrieval, can be adapted
 1394 for memory selection. It scores memory items (treated as terms) based on their frequency in each
 1395 instance relative to the entire dataset, balancing how common a memory item is with how discrimi-
 1396 native it is. Items with higher BM25 scores are considered more informative and are selected as the
 1397 most relevant, making BM25 well-suited for sparse or high-dimensional text and categorical data.

1398 Second, ReContriever (Lei et al., 2023) is a retrieval-based method that scores memory items by
 1399 their ability to help a model retrieve relevant information. It leverages dense embeddings to measure
 1400 semantic similarity between queries and candidate items, effectively identifying the most informa-
 1401 tive items for downstream tasks. Memory items that yield higher retrieval relevance are prioritized,
 1402 making ReContriever particularly effective for selecting contextually important or predictive inputs
 1403 in text or structured datasets. Similarly to semantic similarity a threshold of $t = 0$ is used as thresh-
 old for selecting memory items, with a maximum of up to $n = 5$ selected.

1404 G.5 RUMS-UTILITY AND RUMS-MODEL DETAILS

1405
1406 In this paper, we describe two main methods: RUMS-Utility, which selects a subset of user memory
1407 items with the highest utility for personalization directly, and RUMS-Model, a lightweight trained
1408 model built on data generated by RUMS-Utility.

1409
1410 **RUMS-Utility:** We use RUMS-Utility not only for the analyses in H1 and H2 but also to generate
1411 training data for fine-tuning RUMS-Models. In general, we compute entropy in Equation (6) using
1412 Llama-3.1-8B (AI, 2024b) with a temperature of 0.7 and sampling-based decoding to obtain the
1413 response distribution. We restrict the maximum generation length to $T = 20$ tokens, chosen to
1414 mitigate bias from longer generations having naturally lower entropy.

1415 Following the algorithm in Section 3.1.2, we first prompt GPT-4 to reduce the candidate memory
1416 items from 50 to at least 1 and at most 10 before computing RUMS-Utility (see Prompt 7). Utility
1417 scores are then estimated using $n = 5$ Monte Carlo samples. For each input, we generate $n = 5$
1418 distinct user profiles as separate training points, which can yield entirely unique best memory subsets
1419 of data depending on the values in the user profile.

1420 The chosen hyperparameters (e.g., model size, number of Monte Carlo samples, etc.) reflect a trade-
1421 off: obtaining reliable signals of memory item importance while keeping computations lightweight
1422 and feasible for large-scale inference. Further discussion of these choices is provided in Ap-
1423 pendix B.1.

1424 **Prompt 7.** *Given a user-input and list of features, select which features might be important for*
1425 *answering the user-input.*

1426 *Choose at least 1 and up to 10 features.*

1427 *Respond in a JSON with one key "Chosen Features", which has a list of chosen features. ' Do not*
1428 *include any other text or formatting.*

1430 **Choosing the Personalization Threshold:** To determine the threshold for deciding whether to
1431 apply personalization or select no personalization, we used a held-out validation set of size $n = 100$
1432 from each of PersonaFeedback and FreebaseQA. For each query, we computed the average maxi-
1433 mum utility score over the $n = 5$ generated user profiles. We labeled inputs from PersonaFeedback
1434 as personalized (1) and inputs from FreebaseQA as non-personalized (0). These scores were com-
1435 bined and labeled by dataset to construct an ROC curve.

1436 We then selected the threshold that maximized Youden’s J statistic ($J = \text{TPR} - \text{FPR}$), which iden-
1437 tifies the point that best balances sensitivity and specificity. Using this procedure, we determined an
1438 optimal threshold of $t = 0.29$, which was applied in all downstream analyses.

1440 **RUMS-Models** For training the downstream models, we sampled approximately 2.5K unique
1441 PersonaFeedback queries and 2K unique FreebaseQA queries. Each query was paired with five
1442 user profiles, and RUMS-Utility was applied to select the best memory subsets, yielding 13K and
1443 9K training examples, respectively. We excluded all FreebaseQA queries that exceeded the per-
1444 sonalization threshold, thereby removing any noisy non-personalized samples. This design choice
1445 emphasized that non-personalized inputs should not include any memory items.

1446 We then used these (query, memory subset) pairs to train two models. The first model, used in
1447 H1–H3, was a binary logistic regression implemented with DeBERTa-v3-large (He et al., 2021) and
1448 a final linear classification layer. To train this model, we converted each (query, memory subset) pair
1449 into multiple (query, memory item, label) triplets. Training was performed with a learning rate of
1450 5×10^{-6} for 12 epochs, and the best model was selected based on the highest F1 score on a held-out
1451 validation set ($n = 1K$).

1452 The second model, used for H3, was a multiclass, multilabel logistic regression implemented again
1453 with DeBERTa-v2-large (He et al., 2021) and a final multiclass classification layer. Here, labels
1454 were constructed by merging the selected subsets from the $n = 5$ diverse user profiles per query.
1455 This produced a single label vector in which all memory items chosen in any of the five samples
1456 were marked as significant. This resulted in more dense labels, training the model to select more
1457 memory items. Training was performed with a learning rate of 2×10^{-5} for 12 epochs, and the best
model was again chosen based on the highest F1 score on a held-out validation set ($n = 1K$).

Method	Model	# Parameters
RUMS	microsoft/deberta-v3-large	418M
Semantic Similarity	all-MiniLM-L6-v2	~22.7M
GPT4o	GPT4o	Not disclosed (est. >200B)
ReContriever	Yibin-Lei/ReContriever	~110M

Table 11: Baseline methods and their corresponding model sizes.

Selecting Regression Threshold Since RUMS-Models is trained as a logistic regression model (either binary or multiclass), its output represents the probability of selecting a particular memory item. A common approach is to use a cutoff of 0.5 to make a binary decision, however, since our task is more specialized, instead we tuned this cutoff using a held-out evaluation set ($n = 1K$). Specifically, we perform a grid search over boundaries in $[0, 1]$ and select the cutoff that achieved the highest F1 score on the evaluation set. This yielded $b = 0.81$ for RUMS-Binary and $b = 0.93$ for RUMS-Multi.

G.6 COMPARING SIZE OF METHODS

In Table 11, we summarize the size of each model used in both our method and the baselines. Our approach uses a model comparable in scale to SOTA baselines, yet it is over $400\times$ smaller than GPT-4.

G.7 EVALUATION METRIC AND ANALYSIS

H1 Evaluation: In H1, we compare the distribution of maximum utility scores between assumed personalized and non-personalized inputs. Specifically, we evaluate utility scores on a non-personalized dataset (FreebaseQA) and a personalized dataset (PersonaFeedback). We sampled $n = 500$ inputs from each dataset and computed the maximum utility score using Equation (5) for every input. The resulting distributions were then analyzed using divergence metrics and hypothesis testing.

For divergence metrics, we use Jensen–Shannon (JS) divergence, a symmetric measure of similarity between two probability distributions P and Q . It is defined as:

$$\text{JS}(P\|Q) = \frac{1}{2} \text{KL}(P\|M) + \frac{1}{2} \text{KL}(Q\|M), \quad \text{where } M = \frac{P + Q}{2}$$

Here, $\text{KL}(P\|M)$ is the Kullback–Leibler divergence from P to the average distribution M . JS divergence is always finite, symmetric, and bounded between 0 and $\log 2$.

For hypothesis testing, we use the two-sample Kolmogorov–Smirnov test to assess whether the two samples come from the same distribution. The null hypothesis states that they do, and we test significance at the $\alpha = 0.05$ level.

H1’ Evaluation: In H1’, performance was evaluated using human labels as the gold standard. For each sample, four human annotators rated whether $n = 100$ samples would benefit from personalization using a 0–2 scale. If the annotators’ confidence in personalization was 1 or higher, we considered the input to be personalized. Instructions provided to annotators are given in Prompt 8. We only included inputs with unanimous agreement ($n = 64$) in the evaluation. The average inter-rater agreement between pairs of annotators was $\kappa = 0.59$ using the Cohen’s kappa. The primary metrics used were recall, measuring accuracy in identifying personalized inputs, and specificity, measuring accuracy in identifying non-personalized inputs.

Prompt 8. *Instructions: Read the user input and then decide if the input would benefit from specific user information.*

0 = No Personalization Needed, 1 = Personalize is might be helpful, 2= Personalization is definitely helpful

H2 Evaluation: For H2, we evaluated how closely each method’s selected subset of memory items matched the subsets chosen by humans. To create the human-selected subsets, four annotators were asked to select all memory items (from the full set of 50) that they felt would be helpful in generating a response. Instructions provided to annotators are given in Prompt 9. For each input, we included only the memory items selected by at least three annotators to form the majority subset. This process resulted in an average of 4.19 memory items per query for inputs where at least one item was chosen. This average guided our experimental design for H2 and H3, as we restricted each method to select no more than $n = 5$ items.

We evaluated performance using precision, recall, and F1 score. Here, precision measures the fraction of items selected by a method that were also in the human majority subset, recall measures the fraction of items in the human majority subset that were correctly selected by the method, and F1 is the harmonic mean of precision and recall, providing a single metric that balances both.

	Personalized	Not-Personalized	Difference
Human Annotation	6.45	1.43	5.02
RUMS-Utility	1.87	0.80	1.07
RUMS-Binary	4.87	1.56	3.31
RUMS-Multi	3.27	1.67	1.60
Semantic Similarity	5.00	5.00	0.00
GPT4-ZeroShot	3.30	2.61	0.69
GPT4-FewShot	1.80	1.57	0.23

Table 12: Average number of features selected for personalized vs. not-personalized inputs. Methods closest to human performance are **bolded**.

Number of Memory Items Selected in H2 Lastly, we include table 12 which compares the average number of memory items selected by each method in H2. Of note, we see that human annotations show a clear separation: personalized inputs have on average 6.5 features selected, compared to only 1.4 for non-personalized inputs, a difference of 5. Among automatic methods, RUMS-Model and RUMS-Utility comes closest to this behavior, achieving a range of 3.3 and 1.08, whereas the next closest method is GPT4-ZeroShot which achieves only a 0.67 difference. This further highlights how RUMS excels at both distinguishing when memory items are beneficial as well as which features are beneficial.

Prompt 9. *For each input, decide which (if any) user-features would be helpful to better respond to the input.*

H3 Evaluation: In H3, we evaluate the downstream impact of the memory subsets chosen by each method on generated responses. For each user input, responses are generated using the query combined with the subset of memory items selected by each method. This evaluation is performed with two models: GPT-4 (OpenAI et al., 2024), a high-performing model with robust capabilities, and LLaMA 70B Instruct (AI, 2024a), a smaller model more susceptible to noise. This setup demonstrates RUMS-Models’s flexibility in adapting to models with varying capacities and usage scenarios.

Performance is measured using *generation win rate*, defined as the proportion of pairwise comparisons in which RUMS-Models’s output is judged preferable to that of a baseline. GPT-4 is used as the judge, comparing responses generated with RUMS-Models’s selected subset against those generated with each baseline’s subset. LLM-as-Judge is validated to align closely with third-party human annotations for personalization when combined with in-context examples, model confidence scores (Dong et al., 2024).

To ensure robust evaluation, our methodology includes:

- *In-context examples:* Three examples are provided in the prompt for each evaluation to demonstrate the expected style of reasoning and scoring.
- *Confidence thresholds:* Only responses for which the model reports a self-generated confidence score above the 70% percentile are included in the final analysis.

- 1566
- 1567
- 1568
- 1569
- *Chain-of-thought prompting*: Evaluation prompts encourage the model to reason step-by-step and justify its preference for one response over another, improving consistency and interpretability of the win-rate comparisons.

1570 For personalized datasets (PersonaFeedback and WildChat-Personal), the evaluation metrics in the
1571 chain-of-thought include:

- 1572
- 1573
- 1574
- 1575
- 1576
- 1577
- *Completeness*: Does the response fulfill the user’s request?
 - *Personalization*: Does the response appropriately incorporate relevant details from the user profile?
 - *Overpersonalization*: Does the response include irrelevant, distracting, or unnecessary user details?

1578 For non-personalized datasets (FreebaseQA and WildChat-NonPersonal), the metrics are:

- 1580
- 1581
- 1582
- *Completeness/Accuracy*: Does the response correctly and fully answer the user’s request?
 - *Neutrality*: Is the response unbiased and general, rather than tailored to the user?

1583 Full evaluation prompts can be found in Prompt 11 for personalized datasets, Prompt 12 for non-
1584 personalized datasets, and Prompt 10 for the three in-context examples used in all evaluation.

1585 **Prompt 10. Example 1:**

1586

1587 *”user profile”*: [’Preferred exercise activities: Weightlifting’, ’Hobbies and interests: [’Surfing’,
1588 ’Photography’]’, ’Location (City/State/Country): Melbourne/Victoria/Australia’, ’Social media
1589 platforms used: [’Facebook’, ’LinkedIn’]’, ’Technology usage (devices, apps): [’Smartphone’,
1590 ’Laptop’]],

1591 *”user input”*: ” Why are there five rings in Olympic logo? And what does they symbolize?”,

1592 *”response_1”*: ” The five rings in the Olympic logo represent the five continents: Africa, the Amer-
1593 icas, Asia, Europe, and Oceania. They are interlinked to symbolize the unity and friendship among
1594 nations, highlighting the idea of bringing together athletes from around the world to compete in the
1595 spirit of sportsmanship. The colors of the rings (blue, yellow, black, green, and red) were chosen
1596 because at least one of these colors appears in the national flags of all participating countries.”,

1597 *”response_2”*: “The five rings in the Olympic logo represent the five continents of the world, Africa,
1598 the Americas, Asia, Europe, and Oceania. This design, created by Pierre de Coubertin in 1913,
1599 symbolizes the unity and connection among the nations and cultures participating in the Olympic
1600 Games. The rings are interlocked, indicating that these continents are united through the spirit of
1601 sport, competition, and the Olympic ideals of excellence, friendship, and respect. The colors of the
1602 rings (blue, yellow, black, green, and red) along with the white background were chosen because at
1603 least one of these colors appears on the flag of every nation competing in the Olympics.”,

1604 *“overall”*: “0”,

1605

1606 *“overall_explanation”*: “Although they are worded a bit differently, they generally have the same
1607 information and do not include unhelpful user-information in the answer.”

1608 **Example 2:**

1609 *”user profile”*: [’Social media platforms used: [’Instagram’, ’Facebook’]’, ’Dietary restrictions
1610 (e.g., vegetarian, vegan): None’, ’Gender: Female’, ’Pet ownership (types of pets): [’Cat’]’, ’Cur-
1611 rent projects or goals: Continue education’, ’Religious beliefs: Christian’, ’Favorite TV shows:
1612 [’Grey’s Anatomy’]’, ’Reasons for using the service: Medical updates’, ’Current challenges or
1613 pain points: Shift work’, ’Travel preferences (adventurous, relaxing): Relaxing’],

1614 *”user input”*: ”The 2011 album ’The King of Limbs’ was released by which English rock band?”,

1615

1616 *”response_1”*: ”The 2011 album ’The King of Limbs’ was released by the English rock band Radio-
1617 head.”,

1618 *”response_2”*: “The album ’The King of Limbs’ was released by the English rock band Radiohead”,

1619 *“overall”*: “0”,

1620 “overall_explanation”: “These answer are the same, minus only minor differences”

1621 Example 3:

1622
1623 “user profile”: [‘Current challenges or pain points: Commuting stress’, ‘Location
1624 (City/State/Country): Bangalore/Karnataka/India’, ‘Favorite sports: Cricket’, ‘Personal values or
1625 beliefs: [‘Kindness’, ‘Resilience’]’, ‘Political affiliation: Centrist’, ‘Name: Arjun Mehta’, ‘Reasons
1626 for using the service: Access tools’, ‘Financial situation (broad categories): Financially modest’,
1627 ‘Level of tech-savviness: Moderate’, ‘Cultural background: South Asian’, ‘Frequency of using the
1628 service (daily, weekly): Weekly’, ‘Favorite TV shows: [‘Sacred Games’]’, ‘Travel preferences (ad-
1629 venturous, relaxing): Relaxing’, ‘Preferred mode of communication (text, voice, video): Voice’,
1630 ‘Favorite foods: [‘Dosa’, ‘Chaat’]’, ‘Health and fitness goals: Improve health’],

1631 “user input”: “Are there any recommended effective ways to enhance the fun of English grammar
1632 learning?”

1633 “response_1”: “Hi Arjun! Here are some recommended ways to enhance the fun of learning En-
1634 glish grammar: 1. **Games and Quizzes**: Engage in grammar games or online quizzes that
1635 challenge your understanding in a fun way. Websites like Kahoot! or Quizlet offer interactive tools.
1636 2. **Group Learning**: Form a study group with friends or peers. This can make the learning
1637 process more social and less intimidating. You can discuss grammar topics over chai or dosa! 3.
1638 **Music and Lyrics**: Listen to English songs and try to analyze the lyrics. Identifying verb tenses
1639 or other grammatical structures within your favorite songs can be both educational and entertain-
1640 ing. 4. **Humor in Learning**: Explore humorous content, such as grammar memes or funny
1641 videos about grammar mistakes, to lighten the mood. 5. **Role Play**: Practice dialogues using
1642 correct grammar in fun role-playing scenarios that interest you, perhaps even related to cricket! By
1643 combining these methods with your existing interests and values, like resilience and kindness, you
1644 can make grammar learning a much more enjoyable experience. Happy learning!”

1645 “response_2”: “Improving the enjoyment of English grammar learning can be approached in var-
1646 ious fun and engaging ways. Here are some recommendations that might resonate with you: 1.
1647 **Gamify Learning**: Use educational games and apps that focus on grammar. Many platforms
1648 have quizzes, challenges, and rewards that make learning feel like a fun competition. 2. **Inter-
1649 active Activities**: Engage in activities like grammar charades or sentence-building races with
1650 friends or family. This adds a social element to learning. 3. **Threshold Learning**: Set small,
1651 achievable goals and reward yourself when you reach them. This creates a sense of accomplish-
1652 ment and makes the process more enjoyable. 4. **Join a Language Group**: Participate in local
1653 or online language exchange groups where you can practice speaking with others. This can be a
1654 relaxing way to enhance your skills without the pressure of formal study. Would you like any specific
1655 resources or activities tailored to your preferences?”

1656 “overall”: “2”

1657 “overall_explanation”: “Response 1 is over personalized to the profile and inserts user-information
1658 in an unnatural and unhelpful way”

1659 **Prompt 11.** You are an impartial and knowledgeable evaluator. You will be shown a user profile,
1660 a user input, and two model-generated responses. Your task is to decide if **only one** of the re-
1661 sponses **meets** the evaluation criteria. If both responses do (or neither does), mark it as a tie
1662 (0).

1663 Evaluation Criteria:

1664 - **Completeness**: Does the response fulfill the user’s request?

1665 - **Personalization**: Does the response appropriately use relevant details from the user profile?

1666 - **Overpersonalization**: Does the response include irrelevant, distracting, or unnecessary de-
1667 tails about the user?

1668 - **Overall**: Taking into account completeness, and being appropriately personalized without
1669 being overly personal, is one response clearly better than the other?

1670 Important Notes:

1671 - Assign a score of 1 if **only response 1** satisfies the criterion.
1672
1673

- 1674 - Assign a score of 2 if **only response 2** satisfies the criterion.
1675
1676 - Assign a score of 0 if **both responses** satisfy it or **neither does**.
1677 - Do not favor length or style unless it clearly affects the criterion.
1678 - Be strict: if the difference is marginal, mark it as a tie (0).
1679
- 1680 *Output Instructions:*
1681 *Each of the 4 score fields must be:*
1682
1683 - "1" = only response 1 satisfies the criterion
1684 - "2" = only response 2 satisfies the criterion
1685 - "0" = both or neither satisfy the criterion
1686
- 1687 *Additionally, assess your confidence in this decision by assigning a certainty level from 1 to 100.*
1688 *Use the following guidelines to assign the certainty level:*
1689 *1–20 (Uncertain): There is insufficient or minimal evidence. The decision is largely based on weak*
1690 *or indirect hints.*
1691 *21–40 (Moderately Confident): There is noticeable evidence supporting a preference, though it is*
1692 *not comprehensive, and other interpretations are possible.*
1693 *41–60 (Quite Confident): There is clear and convincing evidence that supports this decision, though*
1694 *it is not entirely decisive.*
1695 *61–80 (Confident): There is strong evidence that clearly supports this decision, with very little*
1696 *ambiguity.*
1697 *81–100 (Highly Confident): There is direct and explicit evidence that decisively supports this pre-*
1698 *diction.*
1699
1700
- 1701 *Lastly, include a **brief (1–2 sentences)** and justify the score using the criteria above.*
1702 *You must return a JSON object with these 12 keys:*
1703
1704 - "completeness", "completeness_explanation", "completeness_confidence"
1705 - "personalization", "personalization_explanation", "personalization_confidence"
1706 - "overpersonalization", "overpersonalization_explanation", "overpersonalization_confidence"
1707 - "overall", "overall_explanation", "overall_confidence"
1708
1709 **Do not include any text outside the JSON block.** Here is the evaluation task:
1710
1711 *Given: "user input": "user_input", "response_1": "response_a", "response_2": "response_b"*
1712 *Generation:*
1713
- 1714 **Prompt 12.** *f''You are an impartial and knowledgeable evaluator. You will be shown a user input*
1715 *and two model-generated responses. Your task is to decide if **only one of the responses** meets*
1716 *the evaluation criteria. If both responses do (or neither does), mark it as a tie (0).*
- 1717 *Evaluation Criteria:*
1718 - **Completeness & Accuracy**: *Does the response correctly and fully answer the user's request?*
1719 - **Neutrality**: *Is the response unbiased and general, rather than tailored to the user?*
1720 - **Overall**: *Taking into account completeness, accuracy, and neutrality is one response clearly*
1721 *better than the other?*
1722
- 1723 *Important Notes:*
1724
1725 - Assign a score of 1 if **only response 1** satisfies the criterion.
1726 - Assign a score of 2 if **only response 2** satisfies the criterion.
1727 - Assign a score of 0 if **both responses** satisfy it or **neither does**.

1728 - Do not favor longer responses unless they add essential information.
1729
1730 - Be strict: if the difference is marginal, mark it as a tie (0).
1731 *Output Instructions: Each of the 4 score fields must be:*
1732 - "1" = only response 1 satisfies the criterion
1733
1734 - "2" = only response 2 satisfies the criterion
1735
1736 - "0" = both or neither satisfy the criterion
1737 *Additionally, assess your confidence in this decision by assigning a certainty level from 1 to 100.*
1738 *Use the following guidelines to assign the certainty level:*
1739 *1–20 (Uncertain): There is insufficient or minimal evidence. The decision is largely based on weak*
1740 *or indirect hints.*
1741 *21–40 (Moderately Confident): There is noticeable evidence supporting a preference, though it is*
1742 *not comprehensive, and other interpretations are possible.*
1743
1744 *41–60 (Quite Confident): There is clear and convincing evidence that supports this decision, though*
1745 *it is not entirely decisive.*
1746
1747 *61–80 (Confident): There is strong evidence that clearly supports this decision, with very little*
1748 *ambiguity.*
1749
1750 *81–100 (Highly Confident): There is direct and explicit evidence that decisively supports this pre-*
1751 *diction.*
1752 *Lastly, include a **brief** (1–2 sentences) and justify the score using the criteria above.*
1753 *You must return a JSON object with these 12 keys:*
1754 - "completeness_accuracy", "completeness_accuracy_explanation", "complete-
1755 ness_accuracy_confidence"
1756 - "neutrality", "neutrality_explanation", "neutrality_confidence"
1757 - "overall", "overall_explanation", "overall_confidence"
1758 ***Do not include any text outside the JSON block.***
1759
1760 *Here is the evaluation task:*
1761 *Given: "user_input": "user_input", "response_1": "response_a", "response_2": "response_b"*
1762 *Generation:*
1763
1764

1765 H OTHER DETAILS

1767 H.1 SOFTWARE

1768
1769 We used Python 3.13.2, Pytorch 2.7.0, and HuggingFace Transformers 4.52.4. All code is licensed
1770 under the Apache License 2.0.
1771

1772 H.2 HARDWARE

1773
1774 All experiments were run on a cluster with 4 NVIDIA H100 GPUs and 1 NVIDIA A100 GPU
1775 with 80B memory. For most inference jobs we used one GPU. For supervised fine-tuning, we used
1776 one GPU. The training for twelve epochs took about 2 hours.
1777

1778 H.3 ARTIFACT TERMS OF USE

1779
1780 PersonaFeedback(Tao et al., 2025): Apache-2.0 FreebaseQA (Jiang et al., 2019): Creative Commons
1781 Attribution-NonCommercial-ShareAlike 3.0 International License WildChat (Zhao et al., 2024):
AI2 ImpACT License

1836
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889

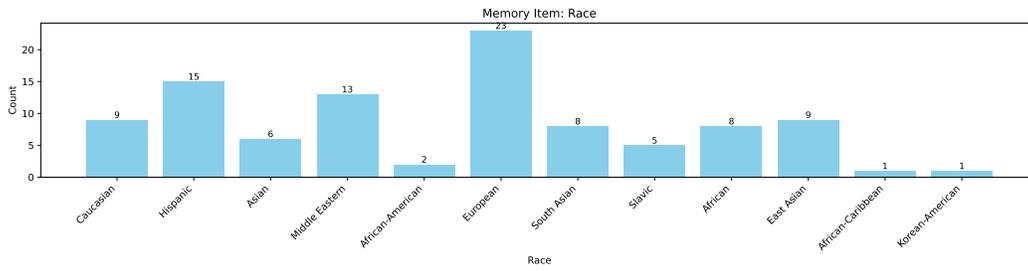


Figure 15: Distribution of Race

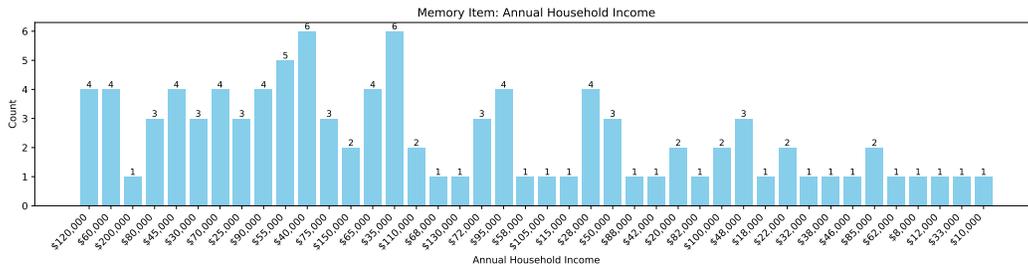


Figure 16: Distribution of Annual Household Income

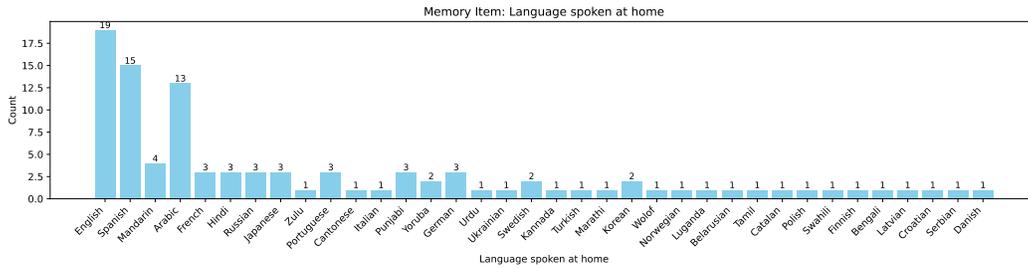


Figure 17: Distribution of Language spoken at home

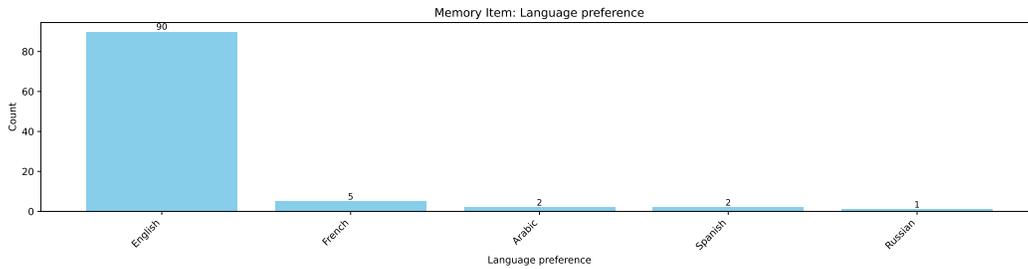


Figure 18: Distribution of Language preference

2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2100
2101
2102
2103
2104
2105

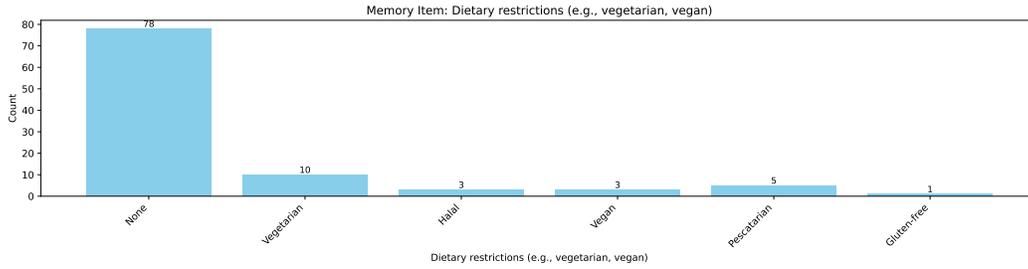


Figure 31: Distribution of Dietary restrictions (e.g., vegetarian, vegan)

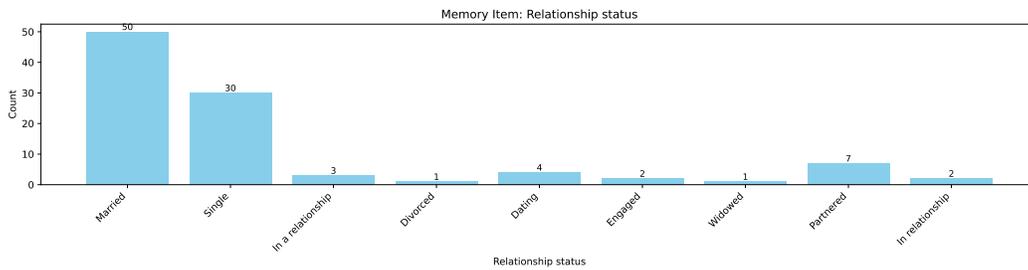


Figure 32: Distribution of Relationship status

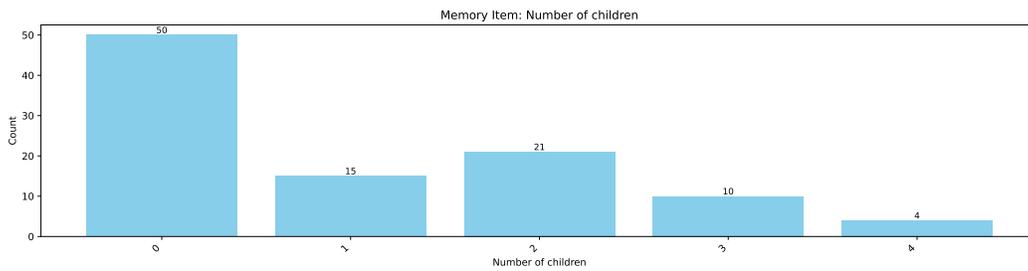


Figure 33: Distribution of Number of children

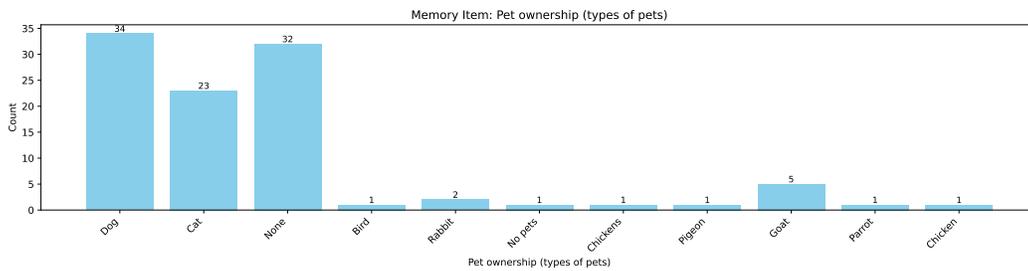


Figure 34: Distribution of Pet ownership (types of pets)

2214
2215
2216
2217
2218
2219
2220
2221
2222
2223
2224
2225
2226
2227
2228
2229
2230
2231
2232
2233
2234
2235
2236
2237
2238
2239
2240
2241
2242
2243
2244
2245
2246
2247
2248
2249
2250
2251
2252
2253
2254
2255
2256
2257
2258
2259
2260
2261
2262
2263
2264
2265
2266
2267

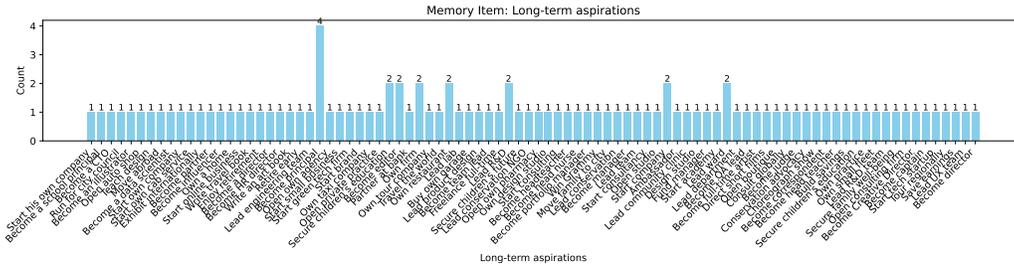


Figure 43: Distribution of Long-term aspirations

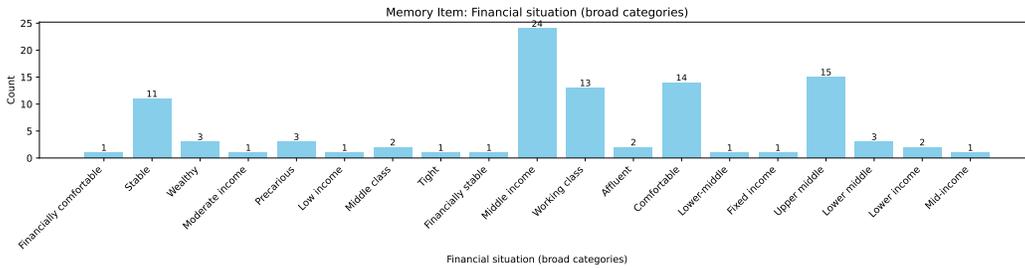


Figure 44: Distribution of Financial situation (broad categories)

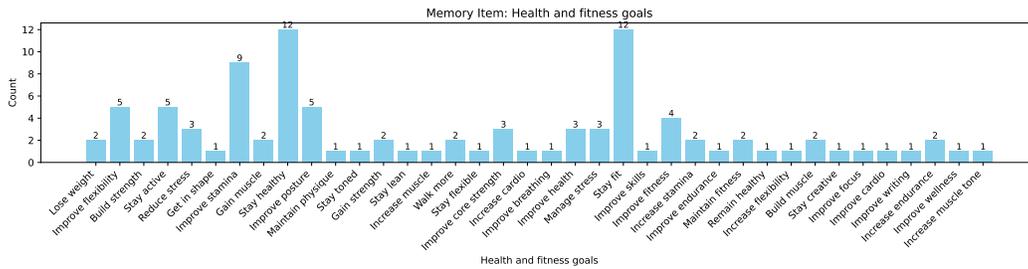


Figure 45: Distribution of Health and fitness goals

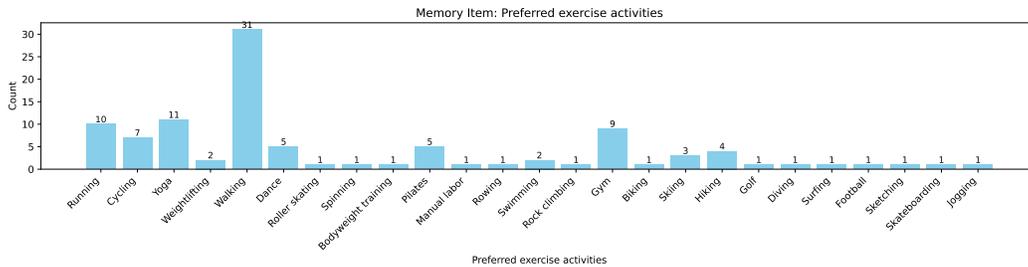


Figure 46: Distribution of Preferred exercise activities

