Tokenize Once, Recommend Anywhere: Unified Item Tokenization for Multi-domain LLM-based Recommendation

Yu Hou[1], **Won-Yong Shin**[1*]
[1]Yonsei University
{houyu, wy.shin}@yonsei.ac.kr

Abstract

Large language model (LLM)-based recommender systems have achieved high-quality performance by bridging the discrepancy between the item space and the language space through item tokenization. However, existing item tokenization methods typically require training separate models for each item domain, limiting generalization. Moreover, the diverse distributions and semantics across item domains make it difficult to construct a unified tokenization that preserves domain-specific information. To address these challenges, we propose **UniTok**, a **Uni**fied item **Tok**enization framework that integrates our own mixture-of-experts (MoE) architecture with a series of codebooks to convert items into discrete tokens, enabling scalable tokenization while preserving semantic information across *multiple item domains*. Specifically, items from different domains are first projected into a unified latent space through a shared encoder. They are then routed to *domain-specific* experts to capture the *unique* semantics, while a *shared* expert, which is always active, encodes common knowledge transferable across domains. Additionally, to mitigate *semantic imbalance* across domains, we present a mutual information calibration mechanism, which guides the model towards retaining similar levels of semantic information for each domain. Comprehensive experiments on wide-ranging real-world datasets demonstrate that the proposed UniTok framework is **(a) highly effective:** achieving up to 51.89% improvements over strong benchmarks, **(b) theoretically sound:** showing the analytical validity of our architectural design and optimization; and **(c) highly generalizable:** demonstrating robust performance across diverse domains without requiring per-domain retraining, a capability not supported by existing baselines.

Code — https://github.com/jackfrost168/UniTok

Introduction

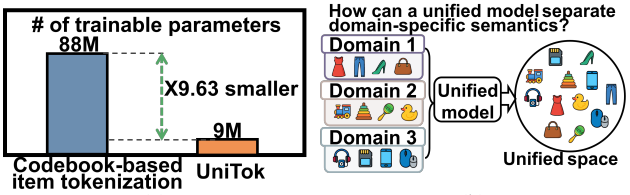
Large language models (LLMs) have recently become a promising paradigm for generative recommendation (Rajput et al. 2023; Hua et al. 2023), leveraging their strong generalization, language understanding, and world knowledge to support personalized recommendation beyond traditional language processing tasks. To effectively use LLMs for recommendation, items must be indexed using identifiers, a process known as item tokenization (Rajput et al. 2023). Item tokenization converts items into *discrete tokens*, such as ID-based representations (Hua et al. 2023), textual descriptors (Zhang et al. 2021), or codebook-based identifiers (Rajput et al. 2023). This bridges the gap between the item space and the language space, enabling LLMs to process items as part of natural language sequences and making generative recommendations feasible.

Existing item tokenization methods (Rajput et al. 2023; Wang et al. 2024) are primarily tailored to items in single-domain settings, necessitating the training of separate tokenizers for each item domain (hereafter, "domain" refers to "item domain" for brevity). In practice, this domain-specific design aligns with the fact that recommender systems are often deployed independently per domain; thus, users rarely perceive quality issues. However, as recommendation tasks increasingly span multiple domains, such as diverse item categories or services, this siloed approach leads to inefficiencies in training, deployment, and maintenance, ultimately hindering scalability. In contrast, other machine learning fields have seen a growing shift towards building *unified models* for multi-domain learning, driven by the need to reduce redundant training, improve parameter efficiency, and facilitate knowledge sharing across domains. Notable advances in this direction have been achieved in language processing (Gururangan et al. 2020; Raffel et al. 2020) and computer vision (Ullah et al. 2022; Jain et al. 2023), demonstrating the feasibility and value of such generalization.

Inspired by this, a natural question arising is: "How can we design a unified item tokenization framework for LLM-based recommendation that can be effectively generalized across multiple domains with minimal computational overhead?" To answer this question, we would like to outline the following two design challenges:

- **C1. Training overhead**: *Repeatedly training* domain-specific tokenizers is inefficient and resource-intensive. As shown in Figure 1a, when applied to 10 distinct domains, our UniTok method reduces the total number of trainable parameters by 9.63× compared to codebook-based item tokenization methods (Rajput et al. 2023; Wang et al. 2024), which require training a separate set of codebooks for each dataset to quantize items.
- **C2. Semantic alignment**: The tokenizer must capture

*Corresponding Author.



(a) (b)

Figure 1: Examples illustrating (a) a comparison of the number of trainable parameters between codebook-based item tokenization methods and our method, UniTok, when applied to 10 distinct item domains, and (b) the inherent challenge of item tokenization across multiple domains.

rich semantics from diverse domains; however, naïvely using a shared token space across domains can cause semantic mixing and biased token assignments. Figure 1b exemplifies this challenge.

To address these aforementioned challenges, we make the first attempt towards developing a **Uni**fied item **Tok**enization framework designed to work effectively across multiple domains, named **UniTok**.

(**Idea 1**): Different domains often have exhibit distinct data distributions, requiring models to be trained separately to capture domain-specific patterns. To move beyond this limitation, we aim to design a unified item tokenization model capable of handling multiple domains without losing domain-specific knowledge. Achieving this goal requires the model to internally disentangle domain-specific learning from shared representations. To this end, we propose a new mixture-of-experts (MoE) architecture, dubbed Token-MoE, wherein domain-specific experts specialize in modeling patterns unique to each domain, while a shared expert captures the common knowledge across multiple domains. This architectural design enables the unified model to retain domain specialization without sacrificing global knowledge sharing (solving **C1** and partially contributing to **C2**).

(**Idea 2**): To preserve item semantics during tokenization, we adopt a codebook-based approach (Rajput et al. 2023) embedded within our MoE architecture, allowing each expert to specialize in distinct semantic patterns. However, the central challenge in multi-domain settings lies in ensuring semantic balance across diverse domains (Ma et al. 2022). To address this, we introduce a mutual information (MI) calibration mechanism that explicitly encourages latent embeddings from each domain to retain sufficient and consistent informativeness. By minimizing the variance of MI across item domains, this mechanism attenuates inter-domain performance variability, enabling more stable and consistent generalization across diverse semantic spaces (solving **C2**).

Our main contributions are summarized as follows:

- **New methodology:** We propose UniTok, a unified item tokenization framework that integrates our customized MoE with codebooks to extract the semantic tokens for items across multiple domains while maintaining semantic balance.
- **Extensive evaluations:** Through comprehensive experimental evaluations on diverse real-world datasets, we demonstrate (a) the superiority of UniTok in multi-domain scenarios, achieving substantial improvements of up to 51.89% in NDCG@10, (b) the efficiency of UniTok, with a 9.63× reduction in model size compared to competitors, and (c) the strong generalization capability of UniTok, exhibiting robust performance in zero-shot settings without additional retraining.
- **Theoretical justifications:** We theoretically prove that UniTok (a) induces a higher entropy token space, (b) achieves a lower quantization error, and (c) ensures semantic consistency across domains by reducing the MI variance, fostering stable and balanced performance.

We refer readers to the supplementary materials for technical details omitted due to page limits.

Preliminaries

Mixture-of-Experts

The mixture-of-experts (MoE) architecture (Jacobs et al. 1991; Shazeer et al. 2017; Fedus, Zoph, and Shazeer 2022; Dai et al. 2024) consists of multiple expert networks, each specializing in handling different parts of the input space. Formally, given an input $\mathbf{x}$, the output of an MoE model is a weighted combination of expert modules: $\text{MoE}(\mathbf{x}) = \sum_{k=1}^{K} G_k(\mathbf{x}) E_k(\mathbf{x})$, where K is the number of experts, $E_k(\mathbf{x})$ denotes the output of the k-th expert, and $G_k(\mathbf{x})$ is a softmax-based router function that assigns a probability to the k-th expert for given input. By dynamically routing input to specialized experts, MoE models can achieve greater model capacity and computational efficiency.

The MoE architecture is not only effective for model scaling but also well-suited for training on a large mixture of datasets (Jain et al. 2023). Its gating mechanism enables adaptive expert selection for each dataset, allowing the model to generalize across diverse sources while minimizing interference between distributions.

Codebook-based Identifiers

Codebook-based identifiers (Rajput et al. 2023) are generated using residual quantization (RQ), which encodes item metadata into hierarchical code sequences by sequentially applying codebooks to residuals across multiple stages. Given an input vector $\mathbf{x}$, the process starts with an initial residual $\mathbf{r}^{(0)} = \mathbf{x}$, and RQ recursively quantizes it using a series of L codebooks $\{C_1, C_2, \ldots, C_L\}$, where $C_\ell \triangleq \{\mathbf{c}_t\}_{t=1}^{T}$ contains T code vectors at level ℓ. The process is defined as

$$\mathbf{c}_\ell = \arg\min_{\mathbf{c} \in C_\ell} \left\| \mathbf{r}^{(\ell-1)} - \mathbf{c} \right\|^2, \tag{1}$$

$$\mathbf{r}^{(\ell)} = \mathbf{r}^{(\ell-1)} - \mathbf{c}_\ell, \tag{2}$$

where $\mathbf{r}^{(\ell)}$ is the residual at level l, and $\mathbf{c}_\ell$ is the nearest code vector selected from codebook C_ℓ. The final approximation of $\mathbf{x}$ after L quantization stages is given by $\hat{\mathbf{x}} = \sum_{\ell=1}^{L} \mathbf{c}_\ell$.

Each selected code vector $\mathbf{c}_\ell$ corresponds to a discrete index $z_\ell \in \{1, \cdots, T\}$, resulting in a token sequence $(z_1, \cdots, z_L)$ that compactly represents the original

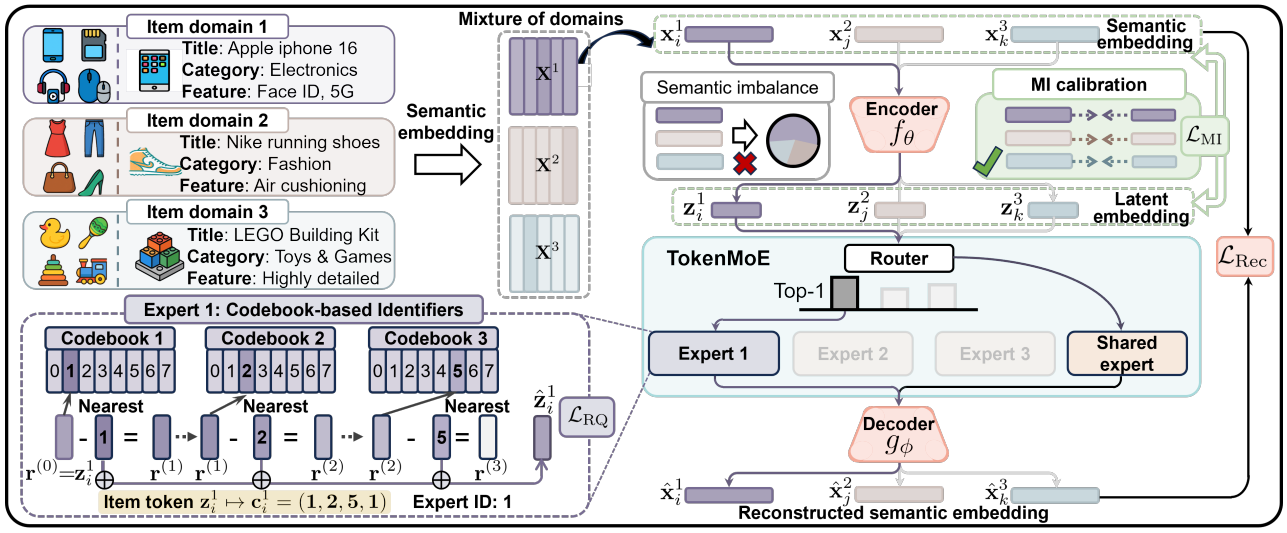


Figure 2: The schematic overview of the proposed UniTok framework.

input. This hierarchical quantization mechanism provides the foundation for our codebook-based identifiers, offering compact and semantically meaningful tokens well-suited for item tokenization. In recommender systems, such discrete tokens can then be directly used as input to LLMs (Rajput et al. 2023; Wang et al. 2024), bridging the gap between item and language spaces while preserving semantic structure.

Methodology

Task Formulation

Multi-domain setting. Let $\mathcal{D} = \{\mathcal{D}_1, \ldots, \mathcal{D}_K\}$ be a mixture of recommendation datasets for K distinct item domains, where each domain $\mathcal{D}_k$ contains an item set $\mathcal{I}_k$ with associated textual metadata, such as titles, categories, features. Unlike typical single-domain scenarios, multi-domain settings pose significant challenges due to distributional inconsistencies (Jain et al. 2023), making it crucial to handle such variability in item tokenization. Addressing this issue is essential for developing a scalable and unified item tokenization method that supports multiple domains, particularly in LLM-based recommender systems. Although collaborative signals can enhance recommendation performance, they inherently rely on user–item interactions, introducing computational overhead and limiting generalization when *shared users are absent* across domains. Instead, we aim to develop a unified tokenization method that operates independently of user data, enabling scalable and general-purpose recommendations in multi-domain LLM-based systems.

Item tokenization task. Given raw items $\mathcal{I}_k$ from any domain-specific dataset $\mathcal{D}_k$, we assume that we have access to a pre-trained content encoder to generate the semantic embeddings for domain k, denoted as $\mathbf{X}^k \in \mathbb{R}^{|\mathcal{I}_k| \times d}$, where $|\mathcal{I}_k|$ is the number of items in domain k and d is the embedding dimensionality. The i-th embedded item is represented as $\mathbf{x}_i^k \in \mathbf{X}^k$. The objective is to learn a mapping function $\mathcal{F} : \mathbb{R}^d \to \mathcal{C}$ that projects each continuous embedding $\mathbf{x}_i^k$ into a discrete codeword $\mathbf{c}_i^k \in \mathcal{C}$, where $\mathcal{C}$ denotes the shared space of discrete item tokens across all domains.

Overview of UniTok

We recall that recent item tokenization methods for LLM-based recommendations focus merely on a single-domain setting (Rajput et al. 2023; Wang et al. 2024; Zheng et al. 2024). Howerer, real-world systems recently operate across multiple domains (Jiang et al. 2022; Ning et al. 2023), leading to repeated training and semantic inconsistency across domains—an issue largely overlooked by prior studies.

To tackle challenges **C1** and **C2** in Section 1, UniTok leverages a shared autoencoder to project items in the mixture of domains into a unified latent space. To achieve effective tokenization across diverse domains, we present TokenMoE with codebook-based identifiers, an MoE architecture composed of domain-specific experts that capture specialized semantics and a shared expert that encodes generalized knowledge. Additionally, we present an MI-based loss that enforces consistent semantics across multiple domains by regulating the informativeness of latent embeddings.

Architectural Details

As illustrated in Figure 2, our UniTok consists of four key components: a shared autoencoder, TokenMoE, codebook-based identifiers, and an MI calibration mechanism.

Shared autoencoder. Given semantic embeddings from the mixture of domains, we first employ a shared autoencoder composed of an encoder f_θ, to project items into a unified latent space, and decoder g_ϕ to reconstruct the semantic embeddings (see the coral pink region in Figure 2). This establishes a unified representation space across multiple domains, which captures common structural patterns while retaining essential information for reconstruction.

Formally, for each input item $\mathbf{x}_i^k \in \mathbf{X}^k$ from domain $\mathcal{D}_k$, the encoder produces a latent embedding $\mathbf{z}_i^k = f_\theta(\mathbf{x}_i^k)$, and

the decoder reconstructs the input item as $\hat{\mathbf{x}}_i^k = g_\phi(\hat{\mathbf{z}}_i^k)$, where $\hat{\mathbf{z}}_i^k = \text{TokenMoE}\left(\mathbf{z}_i^k\right)$ (to be specified in Eq. (5)). The model is optimized using the reconstruction loss:

$$\mathcal{L}_{\text{Rec}} = \sum_{k=1}^{K} \sum_{\mathbf{x}_i^k \in \mathbf{X}^k} \left\| \mathbf{x}_i^k - \hat{\mathbf{x}}_i^k \right\|^2, \tag{3}$$

where $\|\cdot\|$ denotes the L_2 norm. This reconstruction loss shapes the latent space into a compact and informative representation that retains core semantics for item tokenization.

TokenMoE. Conventional tokenization models applied to a unified item space often fail to capture domain-specific nuances, as they treat diverse domains uniformly, potentially leading to the loss of specialized semantic information. To address this limitation, we present TokenMoE, a more generalizable MoE architecture, which routes items to both domain-specific experts and a shared expert. This design enables domain-specific experts to learn specialized patterns, while a shared expert with always-active routing facilitates efficient knowledge transfer across multiple domains. Distinct from earlier approaches that utilize MoE within the feedforward layers of transformers (Dai et al. 2024), our key contribution lies in uniquely integrating MoE into the tokenization module to better enable domain-aware tokenization. The TokenMoE module is illustrated in the light blue region of Figure 2 (with one of the domain-specific expert highlighted in purple and the shared expert in orange). This design addresses **C1** and partially mitigates **C2**.

Specifically, after encoding, the item latent embedding $\mathbf{z}_i^k$ passes through a router function $G(\cdot)$, which produces a softmax distribution over K domain-specific experts: $G(\mathbf{z}_i^k) = \{G_1, G_2, \ldots, G_K\}$, where each G_k is computed as

$$G_k = \frac{\exp(s_i^{(k)})}{\sum_{j=1}^{K} \exp(s_i^{(j)})}, s_i = h(\mathbf{z}_i^k) \in \mathbb{R}^K, \tag{4}$$

and $h(\cdot)$ is a learnable linear transformation in the router producing the router logits s_i, and $s_i^{(k)}$ denotes the logit corresponding to the k-th domain-specific expert. Here, K is the total number of domain-specific experts, which is typically aligned with the number of domains.

The item is then routed to the top-N domain-specific experts[1] based on the highest values of $G(\mathbf{z}_i^k)$, while it is also deterministically assigned to a shared expert. Each expert, including the shared one, is implemented as a codebook-based identifier (to be specified later). The final $\hat{\mathbf{z}}_i^k$ is computed as a weighted combination of the selected domain-specific experts and the shared expert, which is formulated as follows:

$$\begin{cases} \hat{\mathbf{z}}_i^k = \text{TokenMoE}\left(\mathbf{z}_i^k\right) = \sum_{k=1}^{K} G_k E_k(\mathbf{z}_i^k) + E_{\text{share}}(\mathbf{z}_i^k), \\ G_k = \begin{cases} G_k, & \text{if } k \in \text{Top}_N(G(\mathbf{z}_i^k)) \\ 0, & \text{otherwise} \end{cases}, \end{cases} \tag{5}$$

[1] N is typically set as either 1 or 2 to promote sparsity in expert activation, which significantly reduces computational overhead while preserving model capacity (Lepikhin et al. 2021; Fedus, Zoph, and Shazeer 2022).

where, $E_k(\cdot)$ denotes the k-th expert module, $E_{\text{share}}(\cdot)$ denotes the shared expert module, and $\text{Top}_N\left(G\left(\mathbf{z}_i^k\right)\right) = \{k_1, k_2, \ldots, k_N\}$ is the set of indices corresponding to the top-N experts selected by the router. Figure 2 shows an example where the top-1 expert is selected.

TokenMoE routes each item to domain-specific experts via its learned router, while a shared expert captures common knowledge through a deterministic path. To encourage expert specialization, each expert is initialized with the mean feature of a specific domain (Wang et al. 2024), providing a strong inductive bias for domain-aware tokenization *without requiring explicit supervision*. By activating only a subset of experts per item, TokenMoE enhances scalability, maintains domain specificity, and supports better generalization.

Codebook-based identifiers. We adopt RQ (Rajput et al. 2023) in each expert to discretize each item into compact token sequences. Given an item latent embedding $\mathbf{z}_i^k \in \mathbb{R}^d$ produced by the shared encoder, RQ approximates it through a sequence of codebooks $\{C_1, C_2, \ldots, C_L\}$, where L is the codebook size, and each codebook $C_\ell \triangleq \{\mathbf{c}_t\}_{t=1}^{T}$ contains T code vectors. As shown in the bottom-left of Figure 2, at each level ℓ, the residual $\mathbf{r}^{(\ell)}$ from the previous step is encoded using the nearest code $\mathbf{c}_\ell$ in C_ℓ. The sum of all selected codes reconstructs the original latent embedding:

$$E_k(\mathbf{z}_i^k) \approx \sum\nolimits_{\ell=1}^{L} \mathbf{c}_\ell, \quad \text{where } \mathbf{c}_\ell \in C_\ell. \tag{6}$$

This hierarchical quantization enables fine-grained and memory-efficient tokenization, as each item is tokenized by a discrete codeword:

$$\mathbf{z}_i^k \mapsto \mathbf{c}_i^k = (z_1, \ldots, z_L, e_1, ..., e_N), \tag{7}$$

where $z_\ell \in \{1, \ldots, T\}$ denotes the index of selected code vector $\mathbf{c}_\ell$ from the ℓ-th codebook C_ℓ and $e_n \in \{1, \ldots, K\}$ indicates the expert ID of the n-th top-N expert chosen by the router.

To train this quantization process, we adopt the RQ loss:

$$\mathcal{L}_{\text{RQ}} := \sum\nolimits_{\ell=1}^{L} \left\| \text{sg}[\mathbf{r}^{(\ell)}] - \mathbf{c}_\ell \right\|^2 + \alpha \left\| \mathbf{r}^{(\ell)} - \text{sg}[\mathbf{c}_\ell] \right\|^2, \tag{8}$$

where $\mathbf{r}^{(\ell)}$ is the residual vector at level ℓ, $\mathbf{c}_\ell$ is the selected code vector from C_ℓ, $\text{sg}[\cdot]$ is the stop-gradient operator, and α is a balancing hyperparameter. In Eq. (8), the first term aligns the code vector to the target residual (codebook learning), and the second term forces the encoder and router to commit to the selected quantized code vector.

MI calibration. As the shared encoder will project all items into a unified latent embedding space, it is not straightforward for the model to precisely capture domain-specific features due to semantic imbalance. This occurs when the quality of learned latent embeddings varies significantly across diverse domains—particularly between simple and complex domains—causing semantically similar items to be assigned inconsistent tokens (Ma et al. 2022).

As another key contribution aimed at mitigating this issue, we introduce an MI mechanism to ensure that the latent space retains sufficient information from each domain.

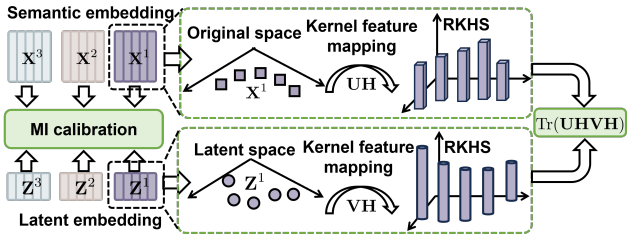


Figure 3: The illustration of MI calibration.

Specifically, we adopt the Hilbert-Schmidt independence criterion (HSIC) (Gretton et al. 2005; Li et al. 2021)[2], serving as a proxy for the MI, with a higher HSIC value indicates stronger dependence. As illustrated in Figure 3, for domain k, HSIC measures the dependence between the input semantic embeddings $\mathbf{X}^k = \{\mathbf{x}_1^k, \ldots, \mathbf{x}_{|\mathcal{I}_k|}^k\}$ and their latent embeddings $\mathbf{Z}^k = \{\mathbf{z}_1^k, \ldots, \mathbf{z}_{|\mathcal{I}_k|}^k\}$ in a reproducing kernel Hilbert space (RKHS), computed as:

$$\widehat{\mathrm{HSIC}}(\mathbf{X}^k, \mathbf{Z}^k) = \frac{1}{(|\mathcal{I}_k| - 1)^2} \mathrm{Tr}(\mathbf{UHVH}), \tag{9}$$

where $\mathbf{U}, \mathbf{V} \in \mathbb{R}^{|\mathcal{I}_k| \times |\mathcal{I}_k|}$ are Gaussian kernel matrices computed over $\mathbf{X}^k$ and $\mathbf{Z}^k$, respectively[3]. The centering matrix $\mathbf{H} = \mathbf{I} - \frac{1}{|\mathcal{I}_k|}\mathbf{1}\mathbf{1}^\top$ ensures zero-mean embeddings in RKHS; and $\mathrm{Tr}(\mathbf{UHVH})$ computes the Hilbert-Schmidt norm of the cross-covariance between the two RKHSs. This design addresses **C2**.

To enforce semantic balance across multiple domains (see Figure 3), we characterize the MI calibration loss as:

$$\mathcal{L}_{\mathrm{MI}} = \mathrm{Var}\left[\widehat{I}^{(k)}\right] - \beta \mathbb{E}\left[\widehat{I}^{(k)}\right], \tag{10}$$

where $\widehat{I}^{(k)} = \widehat{\mathrm{HSIC}}(\mathbf{X}^k, \mathbf{Z}^k)$ and β is a weighting hyperparameter. The first term penalizes high variance of MI across domains to mitigate semantic imbalance, while the second term enforces each domain to retain sufficient domain-specific information.

Optimization. We train the model using the overall loss:

$$\mathcal{L}_{\mathrm{total}} = \mathcal{L}_{\mathrm{Rec}} + \lambda_{\mathrm{RQ}}\mathcal{L}_{\mathrm{RQ}} + \lambda_{\mathrm{MI}}\mathcal{L}_{\mathrm{MI}}, \tag{11}$$

where λ_{RQ} and λ_{MI} are hyper-parameters to control the strength of RQ and MI, respectively.

To instantiate UniTok in LLM-based recommender systems, we first train UniTok on all items using Eq. (11). The trained UniTok then tokenizes each item into discrete semantic tokens, enabling LLMs to operate in the token space. Following prior work (Wang et al. 2024), user interaction histories $\mathbf{u}$ are converted into item token sequences $\mathbf{u} = [\tilde{\mathbf{c}}_1, \tilde{\mathbf{c}}_2, \ldots, \tilde{\mathbf{c}}_P]$, and the LLM-based recommender system learns to predict the next interacted item token $\tilde{\mathbf{c}}_{P+1}$.

[2]Other methods, such as MINE (Belghazi et al. 2018) and InfoNCE (Oord, Li, and Vinyals 2018), can also estimate MI using neural networks, but require additional training. To maintain simplicity, we adopt HSIC, which is non-parametric.

[3]The kernel matrices are computed element-wise using the Gaussian kernel: $U_{ij} = u(\mathbf{x}_i^k, \mathbf{x}_j^k) = \exp\left(-\|\mathbf{x}_i^k - \mathbf{x}_j^k\|^2/2\sigma^2\right)$ and V_{ij}=$v(\mathbf{z}_i^k, \mathbf{z}_j^k)$=$\exp\left(-\|\mathbf{z}_i^k - \mathbf{z}_j^k\|^2/2\sigma^2\right)$ for the bandwidth σ.

Theoretical Analyses

In this subsection, we establish the following theorems, which analyze and support the effectiveness of the key components within UniTok.

Theorem 1. *The token space induced by UniTok exhibits strictly higher entropy than that of standard codebook-based methods:*

$$H(\mathcal{C}_{\mathrm{UniTok}}) > H(\mathcal{C}_{\mathrm{standard}}), \tag{12}$$

where $\mathcal{C}_{UniTok}$ and $\mathcal{C}_{standard}$ denote the discrete token distributions generated by UniTok and standard codebook-based methods, respectively.

This indicates that judiciously incorporating multiple experts increases the overall entropy, thereby expanding the token space capacity.

Theorem 2. *Let $\mathbb{E}[\mathcal{L}_{UniTok}]$ and $\mathbb{E}[\mathcal{L}_{standard}]$ denote the expected quantization error of UniTok and standard codebook-based methods using a single shared codebook, respectively. Then, the following inequality holds:*

$$\mathbb{E}[\mathcal{L}_{\mathrm{UniTok}}] \leq \mathbb{E}[\mathcal{L}_{\mathrm{standard}}]. \tag{13}$$

This implies that a lower expected quantization error reflects more precise modeling of item tokenization. The TokenMoE framework further reduces this error by leveraging expert specialization, which effectively compensates for domain-specific inaccuracies to some extent, thereby improving quantization quality across diverse domains.

Theorem 3. *Suppose that the loss $\mathcal{L}^{(k)}$ on the k-th domain is Lipschitz-continuous with respect to the informativeness of representations. Then, the performance variability across domains is upper-bounded by the variance of MI:*

$$|\mathcal{L}^{(i)} - \mathcal{L}^{(j)}| \leq C\sqrt{\mathrm{Var}\left[\widehat{I}^{(k)}\right]}, \forall i, j \tag{14}$$

where Var $\left[\widehat{I}^{(k)}\right]$ is the variance of MI estimates across domains and C is a constant.

This implies that reducing MI variance across domains promotes consistent semantic representations, leading to more stable and reliable downstream performance.

We empirically validate each theorem's technical correctness and practical relevance through extensive experiments.

Experimental Evaluation

In this section, we carry out comprehensive experiments to empirically validate the effectiveness of UniTok across multiple domains.

Experimental Settings

Datasets. We conduct our experiments on ten real-world datasets spanning ten domains widely adopted for evaluating the performance of recommendations, which include Beauty, Cellphones, Grocery, Instruments, Office, Pet Supplies, Tools, Toys, Games[4], and Yelp[5]. Each item contains metadata, including title, category, and features. Due to page limitations, we report complete results across all datasets to assess overall performance, while employing three representative datasets for the efficiency and ablation analyses.

[4]https://nijianmo.github.io/amazon/index.html.

[5]https://www.yelp.com/dataset.

Method	Beauty	Cellphones	Grocery	Instruments	Office	Pet Supplies	Tools	Toys	Games	Yelp
MF	0.0369	0.0267	0.0216	0.0710	0.0255	0.0268	0.0169	0.0192	0.0366	0.0144
LightGCN	0.0285	0.0456	0.0357	0.0781	0.0301	0.0289	0.0257	0.0287	0.0417	0.0195
SASRec	0.0314	0.0446	0.0376	0.0609	0.0285	0.0301	0.0234	0.0239	0.0412	0.0183
Bert4Rec	0.0194	0.0268	0.0237	0.0573	0.0274	0.0161	0.0092	0.0177	0.0379	0.0131
P5-TID	0.0255	0.0357	0.0316	0.0721	0.0239	0.0243	0.0198	0.0202	0.0388	0.0154
P5-SemID	0.0304	0.0406	0.0351	0.0730	0.0283	0.0282	0.0237	0.0231	0.0432	0.0188
TIGER	0.0324	0.0446	0.0375	0.0788	0.0295	0.0279	0.0284	0.0268	0.0427	0.0208
LC-Rec	0.0381	0.0458	0.0369	0.0802	0.0311	0.0335	0.0307	0.0279	0.0451	0.0215
LETTER	0.0364	0.0473	0.0392	0.0831	0.0326	0.0307	0.0298	0.0291	0.0469	0.0231
UniTok	**0.0478**	**0.0647**	**0.0533**	**0.0884**	**0.0432**	**0.0496**	**0.0439**	**0.0442**	**0.0476**	**0.0321**
Gain	25.46%	36.78%	35.97%	6.38%	32.52%	48.06%	42.99%	51.89%	1.49%	38.96%

Table 1: Performance comparison among UniTok and recommendation competitors for the ten benchmark datasets in terms of NDCG@10. Here, the best and second-best performers are highlighted by bold and underline, respectively. The improvements are statistically significant on average ($p = 0.0219 < 0.05$) based on paired t-tests over five runs across all datasets.

Competitors. To comprehensively demonstrate the superiority of UniTok, we present nine benchmark recommendation methods, including four widely-used collaborative filtering methods, MF (Rendle et al. 2009), SASRec (Kang and McAuley 2018), LightGCN (He et al. 2020), Bert4Rec (Sun et al. 2019), and five item tokenization-aided recommendations, P5-TID (Hua et al. 2023), P5-SemID (Hua et al. 2023), TIGER (Rajput et al. 2023), LC-Rec (Zheng et al. 2024), and LETTER (Wang et al. 2024).

Performance metrics. Following the full-ranking protocol (He et al. 2020), we rank all non-interacted items for each user and evaluate using Recall@M (R@M) and NDCG@M (N@M), where $M \in \{5, 10\}$.

Implementation details. For item tokenization, we use 4-level codebook-based identifiers, where each codebook comprises 256 code vectors with a dimension of 32. We set λ_{RQ} to 1 and λ_{MI} to 0.03. All experiments are conducted on two NVIDIA RTX 3090 GPUs.

Can One Tokenizer Serve All Domains?

To evaluate the recommendation accuracy of UniTok, we compare it with benchmark item tokenization methods across ten benchmark datasets from diverse domains. Notably, distinct from benchmark methods that train separate tokenizers for each dataset, UniTok trains a *single unified model* that handles all ten datasets jointly. As shown in Table 1, UniTok consistently outperforms competitors across all datasets, validating the effectiveness of our unified tokenization framework, achieving up to 51.89% improvement in terms of NDCG@10 on the Tools dataset. Unlike other approaches that require domain-specific customization, UniTok effectively captures item semantics across diverse domains through a single, shared tokenization process—highlighting the strength and generality of our proposed design.

We observe that item tokenization-aided LLM-based recommender systems, including TIGER, LC-Rec, LETTER, and UniTok, are superior to standard collaborative filtering methods such as MF, LightGCN, SASRec, and Bert4Rec. This improvement benefits from the rich semantic understanding and reasoning capabilities inherent in LLMs, which go beyond conventional user–item interactions. Moreover, integrating carefully designed item tokenization into LLM-based recommender systems yields additional performance gains compared to LLMs that solely on item metadata for tokenization (*e.g.*, P5-TID and P5-SemID). This is because item tokenization serves as a critical bridge between the item space and the language space, allowing the underlying model to represent items in a discrete, language-aligned semantic space that enhances generalization and reasoning.

Module	Codebook-based methods	UniTok
Codebook	0.33M	0.36M
Autoencoder	87.45M	8.75M
Router	–	0.01M
Total	87.78M	9.11M

Table 2: Comparison of the number of trainable parameters. For competitors, we report the total number of trainable parameters accumulated across all ten datasets.

Is UniTok More Efficient than Traditional Tokenizers?

To validate the efficiency of UniTok, we compare the total number of trainable parameters between UniTok and traditional codebook-based competitors, including TIGER, LC-Rec, and LETTER, which share the same underlying architecture. While these competitors require training and storing separate tokenization models for each dataset, UniTok employs a single unified model shared across all datasets. For a fair comparison, we report the cumulative trainable parameter count of the codebook-based competitors over all datasets. As shown in Table 2, UniTok achieves approximately a tenfold reduction in the total number of trainable parameters. This efficiency comes primarily from using the shared autoencoder, while the number of additional trainable parameters introduced by the codebook and Token-MoE router remains negligible. By eliminating the need for

	Beauty		Cellphones		Grocery	
Method	**R@10**	**N@10**	**R@10**	**N@10**	**R@10**	**N@10**
TIGER	0.0499	0.0267	0.0661	0.0342	0.0576	0.0273
LC-Rec	0.0564	0.0302	0.0647	0.0337	0.0584	0.0287
LETTER	0.0528	0.0288	0.0678	0.0363	0.0618	0.0315
UniTok	**0.0934**	**0.0478**	**0.1251**	**0.0647**	**0.1061**	**0.0533**
Gain	65.60%	58.28%	84.51%	78.23%	71.68%	69.21%

Table 3: Performance comparison of UniTok and tokenization competitors under a single unified training setup. Here, the best and second-best performers are highlighted by bold and underline, respectively.

	Clothing		Health		Sports	
Method	**R@10**	**N@10**	**R@10**	**N@10**	**R@10**	**N@10**
TIGER	0.0501	0.0242	0.0677	0.0342	0.0469	0.0228
LC-Rec	0.0527	0.0266	0.0694	0.0358	0.0494	0.0246
LETTER	0.0515	0.0257	0.0717	0.0375	0.0510	0.0265
UniTok	**0.0592**	**0.0288**	**0.0835**	**0.0442**	**0.0591**	**0.0298**
Gain	12.33%	8.27%	16.46%	17.87%	15.88%	12.45%

Table 4: Performance comparison among UniTok and recommendation competitors for the three unseen datasets. Here, the best and second-best performers are highlighted by bold and underline, respectively.

	Beauty		Cellphones		Grocery	
Method	**R@10**	**N@10**	**R@10**	**N@10**	**R@10**	**N@10**
UniTok-1	0.0558	0.0304	0.0702	0.0371	0.0633	0.0342
UniTok-2	0.0896	0.0436	0.1194	0.0606	0.0989	0.0497
UniTok-3	0.0915	0.0457	0.1225	0.0622	0.1044	0.0515
UniTok	**0.0934**	**0.0478**	**0.1251**	**0.0647**	**0.1061**	**0.0533**

Table 5: Ablation study results on the Beauty, Cellphones, and Grocery datasets.

domain-specific tokenization learning, UniTok offers substantial advantages in scalability and deployment efficiency.

We further evaluate the performance of competitors under a single unified training setup, where each competitor is trained jointly across ten datasets using a comparable number of trainable parameters to that of UniTok. As shown in Table 3, the competing methods exhibit substantial performance degradation in this setting compared to their single-domain scenario (see Table 1), primarily due to the difficulty of distinguishing items from different domains when using shared tokenization. In contrast, UniTok maintains consistently superior recommendation performance, achieving up to 84.51% improvement in Recall@10 on Cellphones, while using a *similar trainable parameter budget*. This efficiency stems from its modular TokenMoE architecture, where domain-specific experts learn semantics independently, while sharing a unified token space.

Can UniTok be Generalized to Unseen Domains?

To evaluate the generalization ability of UniTok, we adopt a *zero-shot* setting where our item tokenizer model, UniTok, is trained once on the ten source datasets and is directly tested on unseen datasets from three target domains—Clothing, Health, and Sports—without any additional training or fine-tuning. This setup reflects real-world scenarios where new domains may appear dynamically after the model has been deployed.

As shown in Table 4, UniTok significantly outperforms existing item tokenization-based recommender systems, achieving up to 17.87% improvement in NDCG@10 on Health. While the competitors require retraining on each new dataset to achieve reasonable tokenization results, UniTok maintains robust accuracy without any further adaptation. This demonstrates our model's ability to learn a discrete token space that captures transferable item semantics across diverse domains. The consistently strong zero-shot performance further highlights the robustness and practical utility of our unified tokenization framework in supporting effective generalization across heterogeneous domains.

What Makes UniTok Effective?

To assess the contribution of each component in UniTok, we perform an ablation study by progressively removing or modifying its core modules: the TokenMoE module, the shared expert, and the MI calibration part. UniTok-1 removes the TokenMoE and MI calibration, using only a single set of codebooks without any MoE structure; UniTok-2 keeps TokenMoE, but removes the shared expert and MI calibration; and UniTok-3 only removes the MI calibration. UniTok includes all components. As shown in Table 5, removing any module leads to a noticeable drop in recommendation accuracy, which confirms that the combination of modules is crucial to UniTok's effectiveness. In particular, comparing UniTok-1 and UniTok-2 highlights the importance of the TokenMoE module, which significantly improves performance across all datasets by capturing domain-specific semantics. Additionally, in comparison with UniTok-3, introducing MI calibration further enhances performance by enforcing the learned latent embeddings to retain essential semantic information from each domain.

Conclusions and Outlook

We explored an open yet fundamental challenge in multi-domain LLM-based recommendations by building a unified item tokenization framework. To this end, we proposed UniTok, which integrates a customized MoE architecture with codebooks to generate semantically meaningful tokens across diverse domains. Experiments on wide-ranging datasets showed that UniTok (a) achieves up to 51.89% gains in NDCG@10, (b) reduces trainable parameters by 9.63×, and (c) is theoretically validated with respect to the effectiveness of its individual components. Future work includes extending UniTok into a general-purpose tokenization interface for foundation models in recommendation.

Acknowledgments

This work was supported by the National Research Foundation of Korea (NRF) funded by Korea Government (MSIT) under Grants RS-2021-NR059723 and RS-2023-00220762.

References

Belghazi, M. I.; Baratin, A.; Rajeswar, S.; Ozair, S.; Bengio, Y.; Hjelm, R. D.; and Courville, A. C. 2018. Mutual Information Neural Estimation. In *ICML, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, 530–539.

Dai, D.; Deng, C.; Zhao, C.; Xu, R.; Gao, H.; Chen, D.; Li, J.; Zeng, W.; Yu, X.; Wu, Y.; et al. 2024. DeepSeekMoE: Towards Ultimate Expert Specialization in Mixture-of-Experts Language Models. *arXiv preprint arXiv:2401.06066*.

Fedus, W.; Zoph, B.; and Shazeer, N. 2022. Switch Transformers: Scaling to Trillion Parameter Models with Simple and Efficient Sparsity. *Journal of Machine Learning Research*, 23(120): 1–39.

Gretton, A.; Bousquet, O.; Smola, A.; and Schölkopf, B. 2005. Measuring Statistical Dependence with Hilbert-Schmidt Norms. In *ALT, Singapore, October 8-11, 2005*, 63–77.

Gururangan, S.; Marasovic, A.; Swayamdipta, S.; Lo, K.; Beltagy, I.; Downey, D.; and Smith, N. A. 2020. Don't Stop Pretraining: Adapt Language Models to Domains and Tasks. In *ACL, Virtual Event, July 5-10, 2020*, 8342–8360.

He, X.; Deng, K.; Wang, X.; Li, Y.; Zhang, Y.; and Wang, M. 2020. LightGCN: Simplifying and Powering Graph Convolution Network for Recommendation. In *SIGIR, Virtual Event, July 25-30, 2020*, 639–648.

Hua, W.; Xu, S.; Ge, Y.; and Zhang, Y. 2023. How to Index Item IDs for Recommendation Foundation Models. In *SIGIR-AP, Beijing, China, November 26-28, 2023*, 195–204.

Jacobs, R. A.; Jordan, M. I.; Nowlan, S. J.; and Hinton, G. E. 1991. Adaptive Mixtures of Local Experts. *Neural Computation*, 3(1): 79–87.

Jain, Y.; Behl, H. S.; Kira, Z.; and Vineet, V. 2023. DAMEX: Dataset-aware Mixture-of-Experts for visual understanding of mixture-of-datasets. In *NeurIPS, New Orleans, LA, USA, December 10 - 16, 2023*.

Jiang, Y.; Li, Q.; Zhu, H.; Yu, J.; Li, J.; Xu, Z.; Dong, H.; and Zheng, B. 2022. Adaptive Domain Interest Network for Multi-Domain Recommendation. In *CIKM, Atlanta, GA, USA, October 17-21, 2022*, 3212–3221.

Kang, W.-C.; and McAuley, J. 2018. Self-Attentive Sequential Recommendation. In *ICDM, Singapore, November 17-20, 2018*, 197–206.

Lepikhin, D.; Lee, H.; Xu, Y.; Chen, D.; Firat, O.; Huang, Y.; Krikun, M.; Shazeer, N.; and Chen, Z. 2021. GShard: Scaling Giant Models with Conditional Computation and Automatic Sharding. In *ICLR, Virtual Event, Austria, May 3-7, 2021*.

Li, Y.; Pogodin, R.; Sutherland, D. J.; and Gretton, A. 2021. Self-Supervised Learning with Kernel Dependence Maximization. In *NeurIPS, December 6-14, 2021, Virtual Event*, 15543–15556.

Ma, R.; Tan, Y.; Zhou, X.; Chen, X.; Liang, D.; Wang, S.; Wu, W.; and Gui, T. 2022. Searching for Optimal Subword Tokenization in Cross-domain NER. In *IJCAI, Vienna, Austria, 23-29 July 2022*, 4289–4295.

Ning, W.; Yan, X.; Liu, W.; Cheng, R.; Zhang, R.; and Tang, B. 2023. Multi-Domain Recommendation with Embedding Disentangling and Domain Alignment. In *CIKM, Birmingham, United Kingdom, October 21-25, 2023*, 1917–1927.

Oord, A. v. d.; Li, Y.; and Vinyals, O. 2018. Representation Learning with Contrastive Predictive Coding. *arXiv preprint arXiv:1807.03748*.

Raffel, C.; Shazeer, N.; Roberts, A.; Lee, K.; Narang, S.; Matena, M.; Zhou, Y.; Li, W.; and Liu, P. J. 2020. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *Journal of Machine Learning Research*, 21(140): 1–67.

Rajput, S.; Mehta, N.; Singh, A.; Keshavan, R. H.; Vu, T.; Heldt, L.; Hong, L.; Tay, Y.; Tran, V. Q.; Samost, J.; Kula, M.; Chi, E. H.; and Sathiamoorthy, M. 2023. Recommender Systems with Generative Retrieval. In *NeurIPS, New Orleans, LA, USA, December 10 - 16, 2023*.

Rendle, S.; Freudenthaler, C.; Gantner, Z.; and Schmidt-Thieme, L. 2009. BPR: Bayesian Personalized Ranking from Implicit Feedback. In *UAI, Montreal, QC, Canada, June 18-21, 2009*, 452–461.

Shazeer, N.; Mirhoseini, A.; Maziarz, K.; Davis, A.; Le, Q. V.; Hinton, G. E.; and Dean, J. 2017. Outrageously Large Neural Networks: The Sparsely-Gated Mixture-of-Experts Layer. In *ICLR, Toulon, France, April 24-26, 2017*.

Sun, F.; Liu, J.; Wu, J.; Pei, C.; Lin, X.; Ou, W.; and Jiang, P. 2019. BERT4Rec: Sequential Recommendation with Bidirectional Encoder Representations from Transformer. In *CIKM, Beijing, China, November 3-7, 2019*, 1441–1450.

Ullah, I.; Carrión-Ojeda, D.; Escalera, S.; Guyon, I.; Huisman, M.; Mohr, F.; van Rijn, J. N.; Sun, H.; Vanschoren, J.; and Vu, P. A. 2022. Meta-Album: Multi-domain Meta-Dataset for Few-Shot Image Classification. In *NeurIPS, New Orleans, LA, USA, November 28 - December 9, 2022*.

Wang, W.; Bao, H.; Lin, X.; Zhang, J.; Li, Y.; Feng, F.; Ng, S.-K.; and Chua, T.-S. 2024. Learnable Item Tokenization for Generative Recommendation. In *CIKM, Boise, ID, USA, October 21-25, 2024*, 2400–2409.

Zhang, Y.; DING, H.; Shui, Z.; Ma, Y.; Zou, J.; Deoras, A.; and Wang, H. 2021. Language Models as Recommender Systems: Evaluations and Limitations. In *I (Still) Can't Believe It's Not Better! NeurIPS 2021 Workshop*.

Zheng, B.; Hou, Y.; Lu, H.; Chen, Y.; Zhao, W. X.; Chen, M.; and Wen, J.-R. 2024. Adapting Large Language Models by Integrating Collaborative Semantics for Recommendation. In *ICDE, Utrecht, The Netherlands, May 13-16, 2024*, 1435–1448.