arXiv:2608.19529v1 [cs.CL] 20 Aug 2026

When Machines Speak: A Unified Generative Framework for Integrating Machine-Native Symbols into Pretrained Large Language Models

Su Yan
Google Inc.
sueyan@google.com

Rakesh Iyer
Google Inc.
rni@google.com

Abstract

Many real-world AI systems represent entities, behaviors, and structured information using discrete machine-native symbols rather than natural language. While these representations are compact and preserve task-relevant structure, they lie outside the linguistic token space of pretrained large language models (LLMs), creating a fundamental divide between language modeling and structured prediction. We introduce *UniLang*, a unified generative framework that bridges this divide by extending pretrained LLMs to treat machine-native symbols as first-class generative units alongside natural-language tokens. UniLang expands the LLM's vocabulary and embedding space with grounded machine-native representations, enabling textual and symbolic tokens to be jointly modeled and generated under a single autoregressive objective. This unified interface allows pretrained LLMs to directly operate on machine-native representations without requiring them to be verbalized as natural language or relying on task-specific architectures. We evaluate UniLang on two structurally distinct tasks, *sequential recommendation* and *legal precedent prediction*, spanning different domains and types of structured prediction. Across both tasks, UniLang consistently outperforms strong baselines, demonstrating a path toward extending pretrained LLMs beyond language and using them as a common generative modeling backbone for heterogeneous machine-native representations.

1 Introduction

Large language models (LLMs) have emerged as a general modeling framework across a wide range of applications. Their success stems from a unified autoregressive formulation in which diverse natural-language tasks are expressed through a common vocabulary and generation objective. However, this unified interface is fundamentally linguistic: pretrained LLMs operate over natural-language tokens.

Many modern AI systems, in contrast, represent information using *machine-native symbolic representations* rather than natural language. These representations increasingly take the form of discrete codes produced by vector quantization or related techniques, representing complex entities such as compressed audio signals [37], semantic identifiers in recommendation systems [9, 26], and structured relational information in graph learning [20]. Such representations preserve explicit discrete structure and offer computational efficiency that natural language cannot easily replicate. However, they remain outside the linguistic token space of pretrained LLMs, making them difficult for LLMs to natively model and generate.

This mismatch creates a methodological divide. Existing approaches largely follow one of two paradigms. The first treats *natural language as a universal interface*, converting structured information into textual descriptions so that pretrained LLMs can process it [5, 13, 16, 19, 38, 39]. While effective, verbalization may obscure native structure, introduce representational ambiguity, and

sacrifice machine-level precision. The second operates directly on machine-native representations using task-specific models [14, 26, 36, 40]. Although these approaches preserve structural fidelity, they do not directly leverage the linguistic and world knowledge encoded in a pretrained LLM within the same generative space.

At a deeper level, the limitation is representational. Pretrained LLMs lack a unified interface that allows machine-native symbols to function as *first-class generative units* alongside natural-language tokens. Without such an interface, structured symbolic prediction and language modeling remain separate paradigms.

This observation motivates a simple question:

> *Can pretrained LLMs be extended to directly model and generate machine-native symbols, rather than forcing symbolic information to become natural language or abandoning pretrained language models altogether?*

We answer this question with **UniLang**[1], a unified framework that extends pretrained LLMs to jointly model natural-language tokens and machine-native symbols within a single autoregressive vocabulary and generation objective. Rather than treating machine-native representations as external objects or auxiliary embeddings, UniLang explicitly grounds them into the pretrained LLM representation space, allowing machine-native symbols and natural-language tokens to function as first-class generative units within the same autoregressive framework.

This unified representational interface enables structurally different prediction problems to be expressed under the same modeling framework without task-specific architectures. To demonstrate its generality, we evaluate UniLang on two structurally distinct tasks: *sequential recommendation* and *legal precedent prediction*. These tasks were intentionally selected because they represent heterogeneous structured prediction problems rather than multiple datasets from the same application domain. Across both tasks, UniLang consistently outperforms strong baselines while using the same modeling framework.

We make the following contributions:

- **A unified representational interface.** We formulate structured prediction as typed autoregressive generation over heterogeneous token types, allowing natural-language tokens and machine-native symbols to function as first-class generative units within the same pretrained LLM.
- **Grounded integration of machine-native symbols.** We introduce a vocabulary-extension and contrastive grounding procedure that maps machine-native symbols into the pretrained LLM's representation space, enabling them to be processed and generated alongside natural-language tokens while retaining their discrete structure.
- **Generality across heterogeneous structured prediction tasks.** We demonstrate that the same UniLang framework applies to structurally distinct tasks, including sequential recommendation and legal precedent prediction, without task-specific architectural modifications, and consistently outperforms strong baselines.

2 Background and related work

2.1 Machine-native representation.

Machine-native representations convert natural-language content into compact, structured sequences of discrete symbols optimized for computation. The central idea is to encode rich semantic information in a concise symbolic form that facilitates efficient storage, retrieval, and reasoning.

A well-known example is the use of ICD codes in the medical domain [8], where detailed diagnoses are represented as standardized symbolic identifiers. These codes capture complex, structured information that would otherwise require lengthy textual descriptions. Nowadays, machine-native representations follow the same principle but often are learned automatically from data rather than manually defined.

[1] Code is available at `https://github.com/Stella-S-Yan/llm-internalization`

A variety of approaches have been proposed to generate such representations. Early methods rely on clustering-based discretization, such as hierarchical k-means over embedding spaces [33]. Hashing-based techniques map high-dimensional continuous vectors into compact binary codes [18]. Quantization-based approaches, including subspace quantization via k-means, learn discrete codebooks for vector compression [9, 12]. More recent work employs hierarchical residual quantization models, such as Residual Quantized Variational Autoencoders (RQ-VAE), to produce multi-level discrete codes that preserve semantic structure [20, 26, 36]. Figure 1 illustrates an example where each item (a movie) is associated with a discrete, hierarchical machine-native code generated by RQ-VAE. Details of the code construction procedure are provided in Section 3.1.

2.2 Sequential recommendation.

Sequential recommendation aims to predict the next item a user will interact with given their historical interaction sequence. A wide range of neural architectures have been explored for this task, including convolutional, recurrent, graph-based, and Transformer-based models [3, 7, 15, 17, 30, 32, 40].

Recent work has explored reformulating recommendation as a language modeling problem by converting user, item, and interaction information into unified natural-language sequences and training encoder-decoder models [5]. While this verbalization strategy enables the use of LLMs, flattening structured interactions into text can obscure type information and introduce representational ambiguity.

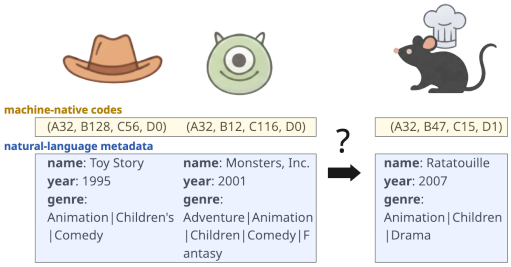


Figure 1: Example of the sequential prediction task. Each item is collectively represented by machine-native codes and various metadata in natural language.

Other approaches encode items as discrete symbols derived from vector quantization of dense embeddings [9]. Extending this idea, [26] generates hierarchical discrete representations using Residual Quantized VAEs (RQ-VAE), referring to the resulting identifiers as Semantic IDs, and integrates them within generative retrieval models. Later work notes that purely discrete codes may lose fine-grained semantic information, and proposes augmenting them with continuous embeddings to recover these signals [36]. While this hybrid approach mitigates information loss, it still treats discrete symbols indirectly within learned embedding spaces rather than as first-class generative units in a unified language modeling framework.

In contrast, UniLang models machine-native symbols and natural-language tokens together as generative units, enabling joint autoregressive prediction of structured and textual content.

2.3 Legal precedent prediction.

Given the context of a legal argument, legal precedent prediction seeks to identify the specific sentence(s) or paragraph(s) from precedential court decisions that supports the claim [22]. This task is substantially more challenging than case-level citation prediction or document retrieval, where the objective is to identify a relevant case as a whole [2, 11]. Legal cases often span tens or hundreds of pages, requiring passage-level methods to perform fine-grained semantic alignment and more involved legal reasoning. Figure 2 illustrates an example citation context and its corresponding target passage from the legal dataset–LePaRD [21].

***Garcia-Giraldo v. United States*, 691 F. Supp. 2d 500 (2010)**

Where, as here, the petitioner has pleaded guilty, the inquiry on collateral review "is ordinarily confined to whether the underlying plea was both counseled and voluntary." United States v. Broce, 488 U.S. 563 (1989)

Figure 2: Example of the legal precedent prediction task. Given context from the **citing opinion** (*Garcia-Giraldo v. United States*), predict the quotation sentence(s) or paragraph(s) from the cited opinion (*United States v. Broce*), which is unknown at inference time.

2.4 Contrastive alignment and token grounding.

To integrate non-linguistic data into language models, the primary challenge is bridging the semantic gap between human vocabulary and specialized machine symbols. Contrastive alignment has been

widely used to ground heterogeneous representations into shared embedding spaces, most prominently in multimodal models such as CLIP [25]. Related techniques have also been explored for grounding new tokens or symbols into pretrained models [4, 31]. Our work differs fundamentally in scope and purpose: rather than maintaining separate representational pathways or learning auxiliary embeddings, we use contrastive grounding as a mechanism to elevate machine-native discrete representations to first-class language tokens. This allows them to be generated, composed, and jointly modeled over within a unified sequence modeling framework.

3 UniLang framework

The core philosophy of UniLang is that natural language and machine-native representations are complementary representations of the same underlying entities. Natural language provides rich, human-interpretable context, while machine-native representations encode compact semantic structures optimized for computation.

Unlike existing approaches that force a choice between these modalities, UniLang treats them as heterogeneous fields within a unified representation and generation space. By enabling joint computation over both types of information, UniLang combines the pretrained linguistic knowledge of LLMs with the compact semantic structure encoded by machine-native symbols.

3.1 Constructing machine-native representations

UniLang is agnostic to the specific mechanism used to generate machine-native representations, requiring only that they consist of discrete, identifiable tokens that can be incorporated into the model vocabulary. In this work, following prior literature on Semantic IDs and residual vector quantization [26, 36], we adopt an RQ-VAE–based discretization pipeline as a concrete instantiation.

Many items of interest admit partial textual descriptions. For example, a movie can be characterized by its title, release year, and genres, while a legal case is naturally described by its textual context, potentially augmented with structured metadata such as court name or case date. A pretrained text encoder maps these descriptions into high-dimensional embeddings, which an RQ-VAE discretizes into a sequence of codes across l quantization levels. Denoting the codebook size at level i by c_i, the resulting representation is $\mathbf{q} = (q^{(1)}, \ldots, q^{(l)}), \quad q^{(i)} \in \{0, \ldots, c_i - 1\}$. Following common practice, we set $l = 3$ and $c_i = 256$ for all levels. To mitigate potential code collisions, we append an additional disambiguation level $q^{(l+1)}$, yielding $(q^{(1)}, \ldots, q^{(l)}, q^{(l+1)})$. For example, $(11, 43, 204, 0)$ and $(11, 43, 204, 1)$ denote two distinct items sharing the same base quantized code. To integrate these discrete codes into the LLM vocabulary, we prepend level-specific prefixes to ensure token-level distinguishability, forming *Semantic IDs* (SIDs) such as (A11, B43, C204, D0). The resulting machine-native vocabulary contains $4 \times 256 = 1{,}024$ tokens, which we call *machine tokens*, capable of representing up to $256^4 \approx 4.3$ billion distinct items. Figure 1 illustrates how items are jointly represented using SIDs and natural-language tokens.

3.2 Vocabulary expansion and semantic grounding

To facilitate seamless modeling of non-linguistic symbols, UniLang establishes a shared vocabulary across modalities. Although pretrained LLMs are optimized for natural language, we extend their latent space with a dedicated set of machine tokens.

Machine tokens are grounded by aligning their embeddings with the textual description embeddings of corresponding items using contrastive learning. Given a pretrained LLM with natural-language vocabulary $\mathcal{V}_{\mathrm{NL}}$, we extend its vocabulary with a fixed set of machine tokens $\mathcal{V}_{\mathrm{ML}}$ (1,024 tokens in our instantiation), each assigned a learnable embedding initialized from a zero-mean normal distribution with standard deviation equal to the model's initializer range.

Let $\mathbf{m}_i = (m_i^1, m_i^2, \ldots, m_i^L) \in \mathcal{V}_{\mathrm{ML}}^L$ denote the machine token sequence representing item i, where L is fixed (e.g. $L = 4$ in our setting). Let $\mathbf{t}_i = (w_i^1, w_i^2, \ldots, w_i^{T_i}) \in \mathcal{V}_{\mathrm{NL}}^{T_i}$ denote the corresponding textual description. Both sequences are encoded by the same pretrained LLM f_θ:

$$\mathbf{z}_i^{\mathrm{ML}} = f_\theta(\mathbf{m}_i), \qquad \mathbf{z}_i^{\mathrm{NL}} = f_\theta(\mathbf{t}_i)$$

The grounding objective encourages the machine token sequence embedding $\mathbf{z}_i^{\mathrm{ML}}$ to align with its corresponding textual description embedding $\mathbf{z}_i^{\mathrm{NL}}$, while remaining distinct from embeddings of other items. Given a batch of N items, we employ an InfoNCE-style contrastive loss [34]:

$$\mathcal{L}_i = -\log \frac{\exp\big(\mathrm{sim}(\mathbf{z}_i^{\mathrm{ML}}, \mathbf{z}_i^{\mathrm{NL}})/\tau\big)}{\sum_{j=1}^{N} \exp\big(\mathrm{sim}(\mathbf{z}_i^{\mathrm{ML}}, \mathbf{z}_j^{\mathrm{NL}})/\tau\big)} \tag{1}$$

where $sim(\mathbf{u}, \mathbf{v}) = \frac{\mathbf{u}^T\mathbf{v}}{\|\mathbf{u}\|\|\mathbf{v}\|}$ denotes cosine similarity and τ is a temperature hyperparameter. To promote bidirectional alignment, we also compute the symmetric loss:

$$\mathcal{L}_i^{sym} = -\log \frac{\exp\big(\mathrm{sim}(\mathbf{z}_i^{\mathrm{NL}}, \mathbf{z}_i^{\mathrm{ML}})/\tau\big)}{\sum_{j=1}^{N} \exp\big(\mathrm{sim}(\mathbf{z}_i^{\mathrm{NL}}, \mathbf{z}_j^{\mathrm{ML}})/\tau\big)} \tag{2}$$

The final grounding objective is given by $\mathcal{L} = \frac{1}{2N}\sum_{i=1}^{N}(\mathcal{L}_i + \mathcal{L}_i^{sym})$.

During grounding, only machine token embeddings are updated; all LLM parameters and natural-language token embeddings remain fixed. Because textual description embeddings are constant, $\mathbf{z}_j^{\mathrm{NL}}$ can be precomputed and reused across training batches, reducing computation. Textual description embeddings are represented using the final hidden state of the sequence, while machine token embeddings are formed via mean pooling over token-level hidden states to ensure equal contribution from each quantization level.

3.3 Unified autoregressive modeling

After grounding machine tokens, we formulate all downstream tasks as sequence-to-sequence generation over heterogeneous token types. Natural-language and machine tokens share a single extended vocabulary $\mathcal{V} = \mathcal{V}_{\mathrm{NL}} \cup \mathcal{V}_{\mathrm{ML}}$ and are modeled autoregressively: for an input sequence $\mathbf{x} = (x_1, \ldots, x_T)$ with $x_t \in \mathcal{V}$, the model defines

$$p_\theta(\mathbf{y}|\mathbf{x}) = \prod_{t=1}^{\|\mathbf{y}\|} p_\theta(y_t|\mathbf{x}, y_{<t}), \quad y_t \in \mathcal{V}$$

No distinction is made between token classes at the modeling level. All tokens share the same embedding table, self-attention layers, and output head, allowing natural-language and machine tokens to interact directly. The framework is task-agnostic. Different downstream tasks are defined by input–output sequence constructions, rather than by changes to the model architecture or training objective.

3.4 Task formulations

To operationalize this framework, we represent tasks as structured hybrid sequences that interleave natural-language tokens and machine-native symbols under a consistent template. Inputs follow a fixed formatting schema that organizes heterogeneous fields in a stable order, while outputs are divided into typed segments using delimiter tokens (e.g., `<year>`, `<genre>`, `<sid>`). These output tags define semantic prediction fields and guide autoregressive generation.

Sequential recommendation. We formulate next-item prediction as typed autoregressive generation over heterogeneous fields. A user's interaction history is encoded as a structured sequence of *SID*, *year*, *genre*, where the *SID* is a machine-native identifier and the remaining attributes are natural-language tokens. Rather than directly predicting a single item ID, the model generates the next item in a field-wise manner: it first produces `<year>` and `<genre>` segments, followed by `<sid>`, the machine-native identifier (Figure 3a).

This structured design discourages degenerate solutions that rely solely on machine-token pattern continuation. Instead, it requires the model to align symbolic sequence prediction with natural-language semantics, coupling structural precision with semantic grounding.

Legal precedent prediction. We apply the same typed generative formulation to legal citation modeling. Citation contexts and quoted passages, together with associated metadata, are represented as structured hybrid sequences. Each text segment is encoded and quantized into a machine-native SID, while metadata fields remain in natural-language form. The input prompt interleaves the context

	template	example
prompt	<sft:think> user {uid}: {{SID},{year},{genre}} prediction:	<sft:think> user 1m_272: A32 B128 C56 D0, 1995, Animation|Children's|Comedy A32 B12 C116 D0, 2001, Adventure|Animation|Children|Comedy|Fantasy prediction:
target	<year>{target_year}</year> <genre>{target_genre}</genre> <sid>{target_SID}</sid>{eos}	<year>2007</year> <genre>Animation|Children|Drama</genre> <sid>A32 B47 C15 D1</sid><|eot_id|>

(a) Sequential prediction

	template	example
prompt	<sft:think> <dcourt>{dest_court}</dcourt> <ddate>{dest_date}</ddate> <dname>{dest_name}</dname> <dsid>{dest_mlid}</dsid> quote:	<sft:think> <dcourt>United States District Court for the Southern District of New York</dcourt> <ddate>2010-03-08</ddate> <dname>Garcia-Giraldo v. United States</dname> <dsid>A1 B37 C190 D20</dsid> quote:
target	<scourt>{source_court}</scourt> <sdate>{source_date}</sdate> <sname>{source_name}</sname> <ssid>{quote_sid}</ssid>{eos}	<scourt>Supreme Court of the United States</scourt> <sdate>1989-01-23</sdate> <sname>United States v. Broce</sname> <ssid>A220 B86 C39 D2</ssid><|eot_id|>

(b) Legal precedent prediction

Figure 3: Examples of unified structured prompting across structurally distinct tasks. Each task uses task-specific metadata and a tailored combination of natural-language and machine tokens, yet all follow the same autoregressive SFT training procedure.

SID with its metadata, and the target sequence consists of the cited passage's metadata followed by its SID (Figure 3b).

This formulation casts citation prediction as structured autoregressive generation over heterogeneous token types. It mirrors the recommendation setting, though it operates over a structurally distinct prediction problem.

4 Experiment

4.1 Datasets

We evaluate UniLang on six real-world benchmarks covering two structurally distinct domains: sequential recommendation and legal precedent prediction. For sequential recommendation, we use the "Beauty" subset of the Amazon Product Reviews dataset [23], along with the MovieLens [6] 1M (ML-1m) and 20M (ML-20m) datasets, which differ substantially in size. For legal precedent prediction, we use LePaRD [21], the largest public dataset for legal case retrieval and prediction, which provides three dataset splits of different sizes. Detailed dataset statistics are provided in Appendix A.

4.2 Evaluation metrics and data splits

We evaluate model performance using Recall@k and NDCG@k. For sequential recommendation, we report results at $k \in 5, 10$, while for legal precedent prediction we use $k \in 1, 10$ to emphasize the importance of top-1 accuracy in legal retrieval tasks. For sequential recommendation, we follow the standard leave-one-out protocol, where the last interaction of each user is used for testing, the second-to-last for validation, and the remaining interactions for training. In legal precedent prediction, we use a 90%/5%/5% split for training, validation, and testing, respectively. To enable efficient hyperparameter tuning, we perform model selection on a random subset of the validation set, while all reported test results are computed on the full test set (see Appendix B for details).

4.3 Machine-native representation generation details

For the Amazon Beauty dataset, we concatenate the raw text from the "title", "categories", "description", "brand", and "price" fields to form a single textual description. For the MovieLens datasets, we combine the "title" with the "genre" field. For the LePaRD dataset, the case context and quoted

precedent passages are directly used as provided. Each resulting text blob is encoded using the open-source `sentence-t5`[2] [24] model to obtain a 768-dimensional dense embedding. We then train an RQ-VAE to quantize these embeddings into discrete machine-native codes (SIDs). Detailed training hyperparameters are provided in Appendix C.

4.4 Machine token grounding implementation details

We use `Llama-3.2-1B-Instruct` [3] as the base model and extend its vocabulary with 1,024 machine-native tokens. The embeddings of these tokens are trained using the contrastive objective described in Section 3.2 (see Table 9 in Appendix for hyperparameters).

Grounding quality is monitored via an alignment metric, computed as the cosine similarity between a machine-token sequence and its corresponding textual embedding. Higher values indicate stronger semantic correspondence. Early stopping is applied based on this metric.

4.5 LLM fine-tuning details

After grounding the machine-native tokens and integrating them into the pretrained `Llama-3.2-1B-Instruct` model, we perform downstream adaptation. During this stage, all original model parameters–including the pretrained token embeddings and the grounded machine-token embeddings–are frozen. We train task-specific Low-Rank Adaptation (LoRA) adapters [10] using supervised fine-tuning (SFT). For sequential recommendation, the maximum user history length is set to 30 for MovieLens and 50 for Amazon Beauty.

For all tasks, LoRA is applied to the projection layers in each transformer block, namely `q_proj`, `k_proj`, `v_proj`, `o_proj`, `gate_proj`, `up_proj`, and `down_proj`. The input–output formatting used for SFT, along with an example prompt and target sequence for MovieLens, is illustrated in Figure 3a. The corresponding formats for LePaRD and Amazon Beauty are shown in Figures 3b and 5, respectively. Complete fine-tuning hyperparameters can be found in Table 10 in the Appendix.

Table 1: Performance comparison on sequential recommendation task.

Dataset	Metric	Discriminative			Generative			Improvement
		SASRec	BERT4Rec	S^3-Rec	P5	TIGER (re-impl.)	UniLang (ours)	
Beauty	Recall@5	<u>0.0387</u>	0.0203	<u>0.0387</u>	0.0163	0.0361	**0.0419**	8.27%
	NDCG@5	<u>0.0249</u>	0.0124	0.0244	0.0107	0.0241	**0.0299**	20.08%
	Recall@10	<u>0.0605</u>	0.0347	**0.0647**	0.0254	0.0595	0.0603	-6.80%
	NDCG@10	0.0318	0.0170	<u>0.0327</u>	0.0136	0.0318	**0.0358**	9.48%
ML-1m	Recall@5	<u>0.1273</u>	0.1011	0.1258	0.1098	0.0634	**0.1661**	30.48%
	NDCG@5	<u>0.0843</u>	0.0649	0.0824	0.0734	0.0412	**0.1145**	35.82%
	Recall@10	0.2013	0.1533	<u>0.2023</u>	0.1575	0.1055	**0.2374**	17.35%
	NDCG@10	<u>0.1083</u>	0.0810	0.1070	0.0888	0.0547	**0.1376**	27.05%
ML-20m	Recall@5	<u>0.0889</u>	0.0616	0.0884	N/A	0.0763	**0.1911**	114.96%
	NDCG@5	<u>0.0549</u>	0.0345	0.0531	N/A	0.0523	**0.1382**	151.73%
	Recall@10	0.1453	0.0948	<u>0.1493</u>	N/A	0.1137	**0.2597**	73.95%
	NDCG@10	0.0728	0.0476	<u>0.0802</u>	N/A	0.0643	**0.1603**	99.88%

Note: The best results are shown in bold, and the second-best results are underlined.

4.6 Performance on sequential recommendation

We first evaluate UniLang on the sequential recommendation task, comparing it against strong task-specific baselines. BERT4Rec [30], SASRec [14], and S^3-Rec [40] are discriminative models specifically designed for sequential recommendation. P5 [5] is a generative approach that verbalizes user, item, and interaction information into natural language, whereas TIGER [26] is also generative but operates purely on machine-native representations (SIDs), predicting the next SID autoregressively.

[2] `https://huggingface.co/sentence-transformers/sentence-t5-base`
[3] `https://huggingface.co/meta-llama/Llama-3.2-1B-Instruct`

Table 2: Run-to-run variability on MovieLens-20M (mean ± SE).

metric	mean ± standard error
Recall@5	0.1908 ± 0.00016
NDCG@5	0.1378 ± 0.00017
Recall@10	0.2596 ± 0.00014
NDCG@10	0.1600 ± 0.00014

For all baselines except TIGER and P5, we report results from the original papers or produce them using the original authors' implementations. For TIGER, we follow the paper and implement the model accordingly. For P5, we adopt corrected and extended results from subsequent works where applicable. Details are provided in Appendix D.

As shown in Table 1, UniLang achieves remarkable performance, outperforming nearly all baselines across benchmarks with up to 151% improvement in NDCG@5 and 115% in Recall@5 on MovieLens-20M. These results demonstrate the effectiveness of extending pretrained LLMs with machine-native symbols: UniLang can directly model and generate structured representations, enabling it to surpass specialized sequential recommendation architectures.

To assess statistical stability, we perform three independent runs with different random seeds on the MovieLens-20M dataset and report mean ± standard error in Table 2.

Table 3: Performance comparison on legal precedent prediction task.

Dataset	**Metric**	**Retrieval**		**Classification**		**Generative**	**Improvement**
		BM25	fine-tuned SBERT	LEGAL-BERT	DistilBERT	UniLang (ours)	
10K	Recall@1	0.0501	0.0899	0.1666	<u>0.1967</u>	**0.2938**	49.36%
	Recall@10	0.1952	0.4479	0.4765	<u>0.5912</u>	**0.6823**	15.41%
	NDCG@10	0.1137	0.2627	0.3075	<u>0.3773</u>	**0.4775**	26.56%
20K	Recall@1	0.0413	0.0753	0.1280	<u>0.1674</u>	**0.2486**	48.51%
	Recall@10	0.1667	0.3850	0.3684	<u>0.5216</u>	**0.6290**	20.59%
	NDCG@10	0.0956	0.2072	0.2377	<u>0.3291</u>	**0.4264**	29.57%
50K	Recall@1	0.0341	0.0500	0.0877	<u>0.1231</u>	**0.1676**	36.15%
	Recall@10	0.1353	0.2590	0.2522	<u>0.3934</u>	**0.4882**	24.10%
	NDCG@10	0.0779	0.1378	0.1625	<u>0.2457</u>	**0.3131**	27.43%

Note: The best results are shown in bold, and the second-best results are underlined.

4.7 Performance on legal precedent prediction

To further demonstrate its generality, we apply the UniLang framework–without any architectural modification–to the structurally different task of legal precedent prediction. Unlike sequential recommendation, which involves modeling chronological sequences of user interactions, legal precedent prediction requires identifying the relevant legal passage from a single citation context, providing a stringent test of UniLang's ability to generalize across fundamentally different problem structures.

We compare UniLang against established retrieval and discriminative baselines. BM25 and fine-tuned SBERT are retrieval-based methods, whereas LEGAL-BERT and DistilBERT approach passage retrieval as a text classification task. All baseline results are taken from the LePaRD paper [21]. Further details on these baselines are provided in Appendix E.

Comparison results are provided in Table 3. Despite the differences in task structure between sequential recommendation and legal passage retrieval, UniLang consistently outperforms all baselines by a substantial margin. In particular, it achieves a 49% improvement in Recall@1 on the 10k dataset, significantly surpassing even specialized discriminative models. These results underscore UniLang's capability to provide a unified modeling framework across tasks with fundamentally different structures.

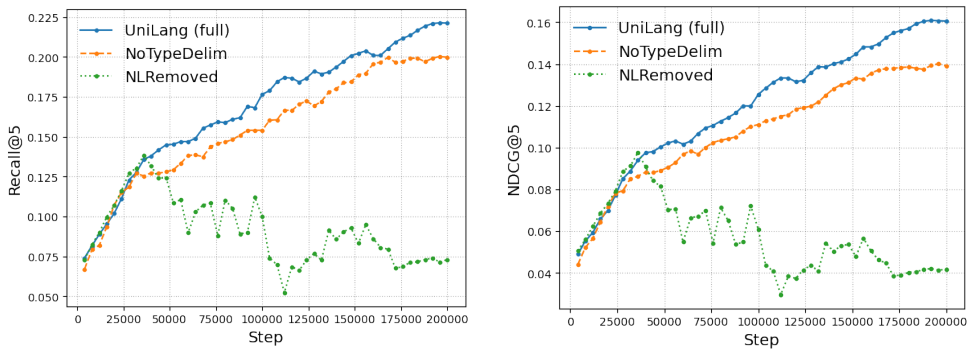


Figure 4: Ablation test on MovieLens-20m.

4.8 Ablation study

We conduct ablation studies on MovieLens-20M to isolate the contributions of semantic grounding, natural-language fields, and structural delimiters. For efficiency, all ablations are evaluated on the fixed validation subset. We consider three variants: (1) NLRemoved, which trains exclusively on machine-native tokens; (2) NoTypeDelim, which removes typed output delimiters (e.g., `<year>`, `</dname>`); and (3) NoWarmup, which uses randomly initialized symbolic embeddings instead of those aligned via contrastive learning.

As shown in Figure 4, NLRemoved exhibits rapid initial improvement followed by performance degradation after 40k steps, suggesting early overfitting when semantic signals from natural language are absent. NoTypeDelim remains stable but consistently underperforms the full UniLang model, indicating that explicit input and output structuring facilitates optimization and improves prediction quality. Notably, NoWarmup fails to achieve non-zero metrics (NDCG and Recall), demonstrating that pre-aligning machine-native symbols to the LLM's latent space is essential for effective generative learning in our setting. We omit this variant from Figure 4 as it remains at the zero-baseline throughout training. Overall, these results suggest that natural language provides regularizing context, while grounded embeddings and typed structures are critical for stability and prediction quality.

	template	example
prompt	<sft:think> user {uid}: {{SID}:{cat}; {brand}} prediction:	<sft:think> user b_10009: A53 B53 C119 D0: Makeup, Face, Concealers & Neutralizers; Maybelline A197 B216 C7 D0: Tools & Accessories, Makeup Brushes & Tools, Brushes & Applicators; Maybelline A198 B110 C88 D0: Makeup, Face, Foundation; Maybelline prediction:
target	<cat>{target_cat}</cat> <brand>{target_brand}</brand> <sid>{target_SID}</sid>{eos}	<cat>Makeup, Eyes, Mascara</cat> <brand>Maybelline</brand> <sid>A35 B21 C172 D0</sid><|eot_id|>

Figure 5: Prompt template and example for the Amazon Beauty dataset.

4.9 Qualitative comparison

So far, our results demonstrate the quantitative benefits of jointly representing natural language and machine-native tokens. We now provide a qualitative comparison with approaches that verbalize all user and item information into natural language, such as P5 [35]. P5 typically requires multiple handcrafted prompts per dataset per task (e.g., 13 prompts for Beauty sequential recommendation). One example prompt is shown below:

> **Input template:** I find the purchase history list of user {{user_id}}: {{history item list of {{item_id}}}}. I wonder which is the next item to recommend to the user?
>
> **Target template:** {{item [item_id]}}

In contrast, UniLang uses a single structured prompt that seamlessly combines natural language with machine tokens, as shown in Figure 5. This unified representation reduces the need for extensive prompt engineering while maintaining or improving performance. By treating language and struc-

tured symbols as complementary generative units, UniLang provides a scalable and generalizable framework for heterogeneous tasks, enabling models to jointly model both token types.

4.10 Emergent representational capabilities

Beyond task performance, UniLang exhibits an important representational advantage. Existing generative recommender systems struggle with user identity modeling. For example, TIGER [26] represents users by hashing raw user IDs into a fixed set of 2,000 ID tokens. As a result, these user ID tokens occupy a large portion of the model's extended vocabulary (66%), which can dominate the token space and limit scalability as the number of users grows.

In contrast, UniLang requires no dedicated user tokens or hashing schemes. User identifiers (e.g., "b_1009") can be directly represented in natural-language form and processed by the pretrained tokenizer. Although not the primary objective of our method, this property highlights UniLang's representational scalability and reinforces its role as a unified modeling framework rather than a task-engineered solution.

5 Conclusion and future work

We introduced *UniLang*, a unified framework that extends pretrained LLMs to model natural language and machine-native symbols within a single autoregressive objective. By treating structured symbols as first-class generative units, UniLang enables structurally diverse tasks–such as sequential recommendation and legal precedent prediction–to be addressed without task-specific architectures. Empirically, UniLang consistently outperforms strong baselines, while ablations demonstrate the importance of natural-language context and typed structure. Overall, UniLang provides a general interface for extending pretrained LLMs beyond language to model heterogeneous machine-native representations.

While UniLang demonstrates strong performance on structured prediction tasks, we did not evaluate the impact of symbolic fine-tuning on the model's natural language generation quality. Our focus in this work is on accuracy in predictive tasks, and studying potential effects on language fluency and coherence is left for future work.

Table 4: Statistics of the sequential recommendation datasets.

Dataset	#users	#items	#actions	Avg. length	Density
Beauty	40,226	54,542	0.35m	8.8	0.02%
ML-1m	6,040	3,416	1m	163.5	4.79%
ML-20m	138,493	26,744	20m	144.4	0.54%

Table 5: Statistics of the legal precedent prediction datasets.

Dataset (# cited passages)	# Citing Passage Surface Forms	# Cited Passage Surface Forms
10k	1,294,730	798,558
20k	1,586,663	1,140,058
50k	2,079,704	1,821,751

A Dataset

- Amazon Beauty [4] [23]: This dataset consists of product reviews collected from Amazon.com. The full collection is organized by top-level product categories, and in our experiments, we use the subset corresponding to the "Beauty" category.
- MovieLens: A widely used benchmark for evaluating recommendation systems. In this study, we use two standard versions: MovieLens 1M (ML-1m) [5] and MovieLens 20M (ML-20m) [6].
- The LePaRD dataset [7] [21] is a large-scale collection of 4.3 million U.S. federal judicial citations. It leverages judges' quotation behavior as supervision, pairing precedential quotations with their corresponding citation contexts. The dataset provides three subsets that differ in the number of top-n most frequently cited passages included. Due to unavoidable preprocessing noise—such as OCR errors, sentence segmentation inconsistencies, and metadata formatting variations—the same citing context or cited quotation may appear in multiple surface forms.

Table 4 presents the statistics of the sequential recommendation datasets. Table 5 provides an overview of the legal precedent prediction dataset. Table 6 summarizes the statistics of the textual features in the legal dataset.

B Validation set sampling

We use Recall@k on the validation set for early stopping during training. Since our framework is generative, we perform beam search to deterministically generate the top-k results and compute Recall@k accordingly. Running beam search over the full validation set is computationally expensive, so to expedite model selection, we instead use a fixed, randomly sampled subset of the validation set generated with a fixed random seed. For all datasets, reported results are based on models selected using this sampled subset. Table 7 lists the sample sizes for each dataset. For all experiments, we use a beam size of 20.

C Model training hyperparameters

All experiments were conducted on 8 NVIDIA H100 GPUs (80GB each). Training can also be performed on a single GPU; multiple GPUs are used primarily to accelerate training via data parallelism. Training time varies depending on batch composition. On the MovieLens-20M dataset,

[4] https://cseweb.ucsd.edu/~jmcauley/datasets/amazon/links.html
[5] https://grouplens.org/datasets/movielens/1m/
[6] https://grouplens.org/datasets/movielens/20m/
[7] https://huggingface.co/datasets/rmahari/LePaRD

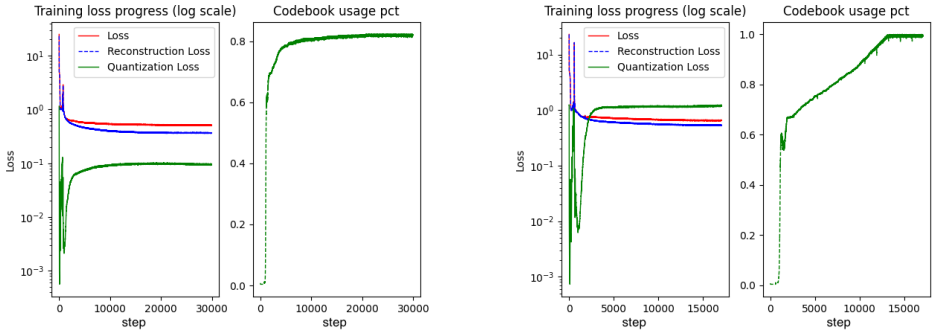
Table 6: Summary statistics of legal precedent dataset text features

Feature	Mean	Std	Min	Max
Length of cited text (chars)	306	225	24	18,342
Length of citing context (chars)	562	216	5	14,062

Table 7: Validation subset sizes for model selection

Dataset	Validation size	Sample size
Beauty	40,226	5,000
MovieLens-1m	6,040	1,000
MovieLens-20m	138,493	1,000
10k	103,812	1,000
20k	134,737	1,000
50k	190,051	1,000

under our configuration, training proceeds at approximately 0.15 seconds per optimization step (batch size 128), with total runtime scaling linearly with the number of training steps.



(a) Beauty (b) LePaRD 20k

Figure 6: RQ-VAE training progress on different datasets.

C.1 RQ-VAE training parameters

The same RQ-VAE architecture is used across all datasets. The model takes 768-dimensional text embeddings as input. The encoder comprises three fully connected layers with output dimensions 512, 256, and 128, each followed by ReLU activation, and outputs a 16-dimensional latent representation. The decoder mirrors the encoder with layers of dimensions 128, 256, and 512. At each quantization level, we maintain a codebook of size 256. Table 8 summarizes the detailed hyperparameter settings, and Figure 6 illustrates the training dynamics of the individual loss components across datasets.

C.2 Machine token grounding parameters

Table 9 summarizes the hyperparameters used for training the embeddings of the machine-native tokens.

C.3 Supervised fine tuning parameters

Table 10 lists the hyperparameters used for supervised fine-tuning of `Llama-3.2-1B-Instruct` on structured input–output sequences containing interleaved natural-language and machine-native tokens.

Table 8: RQ-VAE hyperparameters.

Hyperparameter	Beauty	ML-1M ML-20M	LePaRD 10k / 20k / 50k
Total training steps	30k	20k	20k
Warm up	3k	2k	3k
Peak learning rate	1e-3	1e-3	1e-3
End learning rate	1e-5	5e-5	1e-5
Ema decay	0.99	0.99	0.99
Commitment cost	1.5	1.5	0.1

Note: All experiments use a codebook embedding size of 16 and a batch size of 2048, and are optimized with AdamW (weight decay 0.055, $\beta_1 = 0.9$, $\beta_2 = 0.98$) under a cosine learning rate schedule.

Table 9: Machine token alignment hyperparameters.

Dataset	Beauty	ML-1M ML-20M	LePaRD 10k / 20k / 50k
Batch size	1024	512	128
Learning rate	$1e{-3}$	$1e{-3}$	$6e{-3}$
Total steps	4k	4k	8k

Note: All experiments use the AdamW optimizer (weight decay 10^{-2}, $\beta_1 = 0.9$, $\beta_2 = 0.98$) with a cosine learning rate schedule and 400 warmup steps. The temperature for the contrastive loss is set to 0.2.

D Sequential recommendation baseline methods

- SASRec [14] is a self-attention-based sequential recommendation model that efficiently captures long-term user behavior.
- BERT4Rec [30] trains a bidirectional Transformer with a masked item prediction objective to learn contextualized representations of user behavior sequences.
- S^3-Rec [40] adopts a self-supervised pretraining strategy to capture correlations among items, attributes, and interaction sequences, alleviating data sparsity.
- P5 [5] formulates recommendation as a language modeling problem, representing user–item interactions, user profiles, item metadata, and reviews in natural language.
- TIGER [26] represents items using semantic IDs (SIDs), converting user interaction histories into SID sequences. A generative model is trained from scratch to autoregressively predict the next SID for sequential recommendation.

In evaluating recommendation systems, two protocols are commonly adopted: (1) sampled evaluation, where the ground-truth item is ranked against a fixed set of negative samples (e.g., 99 items), and (2) full-ranking, where the ground-truth item is ranked against all items in the candidate set. While sampled evaluation is computationally efficient, it can lead to overly optimistic performance estimates. In this work, we adopt full-ranking evaluation throughout to provide a more rigorous and realistic assessment of model performance.

For the Amazon Beauty dataset, results for SASRec, BERT4Rec, and S^3-Rec are taken from the publicly released benchmarks [8] provided by the S^3-Rec authors. The P5 results are obtained from the TIGER paper [26], where the authors introduced a preprocessing modification to ensure a fair comparison.

For the MovieLens 1M and 20M datasets, results for SASRec, BERT4Rec, and S^3-Rec are obtained using the authors' released code. All models are trained with their default hyperparameter settings. For SASRec and BERT4Rec, we additionally implement full-ranking evaluation to obtain the reported results.

Results for P5 on the MovieLens 1M dataset are taken from the OpenP5 [35] paper. OpenP5 incorporates the original P5 backbone model, downstream tasks, and item indexing method. Since P5

[8] `https://github.com/RUCAIBox/CIKM2020-S3Rec`

Table 10: SFT hyperparameters.

Dataset	Beauty	ML-1M ML-20M	LePaRD 10k / 20k / 50k
Batch size	8	32/128	512
Learning rate	$1e{-4}$	$2e{-4}$	$2e{-4}$
LoRA dropout	0.25	0.05	0.25
Total training steps	30k	20k	30k
Warm up	2k	2k	2k

Note: All experiments use LoRA with rank 32 and $\alpha = 2$. Optimization is performed using AdamW (weight decay 0.005, $\beta_1 = 0.9$, $\beta_2 = 0.98$) with a cosine learning rate schedule.

requires substantial manual prompt engineering when adapted to new datasets, we consider using OpenP5's reported results to provide a fairer comparison.

Due to the prohibitively high storage and computational cost of applying P5 to the MovieLens 20M dataset, we do not report its results on this dataset (denoted as N/A in Table 1).

E Legal precedent prediction baseline methods

We consider four baseline methods, two retrieval-based and two classification-based.

BM25 represents a sparse lexical retrieval approach [28]. The "fine-tuned SBERT" baseline is a dense embedding-based retrieval method that uses a fine-tuned SBERT [27] model to generate embeddings, followed by maximum dot-product similarity for retrieval.

Passage retrieval can also be formulated as a text classification task, where each target passage is assigned a unique label that serves as the prediction target for its preceding context. Two classification-based baselines are considered: DistilBERT [29] and LEGAL-BERT [1], the latter being a domain-adapted BERT model trained on a large corpus of legal documents. All baseline results are taken from the LePaRD paper [21].

References

[1] Ilias Chalkidis, Manos Fergadiotis, Prodromos Malakasiotis, Nikolaos Aletras, and Ion Androutsopoulos. Legal-bert: The muppets straight out of law school. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2898–2904, Online, 2020. Association for Computational Linguistics.

[2] Faraz Dadgostari, Mauricio Guim, Peter A. Beling, Michael A. Livermore, and Daniel N. Rockmore. Modeling law search as prediction. *Artificial Intelligence and Law*, 29(1):3–34, 2021.

[3] Gabriel de Souza Pereira Moreira, Sara Rabhi, Jeong Min Lee, Ronay Ak, and Even Oldridge. Transformers4rec: Bridging the gap between nlp and sequential/session-based recommendation. In *Proceedings of the Fifteenth ACM Conference on Recommender Systems (RecSys 2021)*, pages 143–153, 2021.

[4] Rinon Gal, Yuval Alaluf, Yuval Atzmon, Or Patashnik, Amit H. Bermano, Gal Chechik, and Daniel Cohen-Or. An image is worth one word: Personalizing text-to-image generation using textual inversion, 2022.

[5] Shijie Geng, Shuchang Liu, Zuohui Fu, Yingqiang Ge, and Yongfeng Zhang. Recommendation as language processing (rlp): A unified pretrain, personalized prompt & predict paradigm (p5). In *RecSys 2022 - Proceedings of the 16th ACM Conference on Recommender Systems*, RecSys 2022 - Proceedings of the 16th ACM Conference on Recommender Systems, pages 299–315. Association for Computing Machinery, Inc, September 2022. Publisher Copyright: © 2022 ACM.; 16th ACM Conference on Recommender Systems, RecSys 2022 ; Conference date: 18-09-2022 Through 23-09-2022.

[6] F. Maxwell Harper and Joseph A. Konstan. The movielens datasets: History and context. *ACM Transactions on Interactive Intelligent Systems*, 5(4):19:1–19:19, Dec 2015.

[7] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. Session-based recommendations with recurrent neural networks, 2015. cite arxiv:1511.06939Comment: Camera ready version (17th February, 2016) Affiliation update (29th March, 2016).

[8] J. A. Hirsch, G. Nicola, G. McGinty, R. W. Liu, R. M. Barr, M. D. Chittle, and L. Manchikanti. Icd-10: History and context. *American Journal of Neuroradiology*, 37(4):596–599, 2016.

[9] Yupeng Hou, Zhankui He, Julian McAuley, and Wayne Xin Zhao. Learning vector-quantized item representation for transferable sequential recommenders. In *Proceedings of The Web Conference 2023*, pages 2808–2818, 2023.

[10] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models, 2021.

[11] Zihan Huang, Charles Low, Mengqiu Teng, Hongyi Zhang, Daniel E. Ho, Mark S. Krass, and Matthias Grabmair. Context-aware legal citation recommendation using deep learning. In *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Law*, ICAIL '21, pages 79–88, New York, NY, USA, 2021. Association for Computing Machinery.

[12] Hervé Jégou, Matthijs Douze, and Cordelia Schmid. Product quantization for nearest neighbor search. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(1):117–128, 2011.

[13] Liangyi Jiang, Xiaocong Liu, Nejatian Nasir-Moin, Hongyi Wang, Abdullah Abidin, Kai Eaton, . . . , and [additional authors]. Health system-scale language models are all-purpose prediction engines. *Nature*, 619:357–362, 2023.

[14] Wang-Cheng Kang and Julian McAuley. Self-attentive sequential recommendation, 2018.

[15] Wang-Cheng Kang and Julian McAuley. Self-attentive sequential recommendation. In *Proceedings of the 2018 IEEE International Conference on Data Mining (ICDM)*, pages 197–206, 2018.

[16] Xiaoyu Kong, Junguang Jiang, Bin Liu, Ziru Xu, Han Zhu, Jian Xu, Bo Zheng, Jiancan Wu, and Xiang Wang. Think before recommendation: Autonomous reasoning-enhanced recommender. In *NeurIPS 2025 Poster Session, San Diego*, 2025.

[17] Jing Li, Pengjie Ren, Zhumin Chen, Zhaochun Ren, Tao Lian, and Jun Ma. Neural attentive session-based recommendation. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management (CIKM 2017)*, pages 1419–1428, 2017.

[18] Fangyuan Luo, Yankai Chen, Jun Wu, Tong Li, Philip S. Yu, and Xue Liu. Learning to hash for recommendation: A survey, 2025.

[19] Yanchen Luo, Junfeng Fang, Sihang Li, Zhiyuan Liu, Jiancan Wu, An Zhang, Wenjie Du, and Xiang Wang. Text-guided small molecule generation via diffusion model. *iScience*, 27(11):110992, 2024.

[20] Yuankai Luo, Hongkang Li, Qijiong Liu, Lei Shi, and Xiao-Ming Wu. Node identifiers: Compact, discrete representations for efficient graph learning. In *The Thirteenth International Conference on Learning Representations*, 2025.

[21] Robert Mahari, Dominik Stammbach, Elliott Ash, and Alex Pentland. Lepard: A large-scale dataset of judicial citations to precedent. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9863–9877, Bangkok, Thailand, 2024. Association for Computational Linguistics.

[22] Robert Zev Mahari. Autolaw: Augmented legal reasoning through legal precedent prediction, 2021.

[23] Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton van den Hengel. Image-based recommendations on styles and substitutes. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 43–52, New York, NY, USA, 2015. ACM.

[24] Jianmo Ni, Gustavo Hernandez Abrego, Noah Constant, Ji Ma, Keith Hall, Daniel Cer, and Yinfei Yang. Sentence-t5: Scalable sentence encoders from pre-trained text-to-text models. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1864–1874, Dublin, Ireland, 2022. Association for Computational Linguistics.

[25] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. *CoRR*, abs/2103.00020, 2021.

[26] Shashank Rajput, Nikhil Mehta, Anima Singh, Raghunandan Hulikal Keshavan, Trung Vu, Lukasz Heldt, Lichan Hong, Yi Tay, Vinh Tran, Jonah Samost, Maciej Kula, Ed Chi, and Maheswaran Sathiamoorthy. Recommender systems with generative retrieval. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 10299–10315. Curran Associates, Inc., 2023.

[27] Nils Reimers and Iryna Gurevych. Sentencebert: Sentence embeddings using siamese bert networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, 2019.

[28] Stephen E. Robertson, Steve Walker, Susan Jones, Micheline Hancock-Beaulieu, and Mike Gatford. Okapi at TREC-3. In *Proceedings of the Third Text REtrieval Conference (TREC 1994)*, Gaithersburg, USA, November 1994.

[29] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: Smaller, faster, cheaper and lighter. In *Proceedings of the 5th Workshop on Energy Efficient Machine Learning and Cognitive Computing (EMNLP 2020 Workshop)*, 2020.

[30] Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. Bert4rec: Sequential recommendation with bidirectional encoder representations from transformer. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management (CIKM 2019)*, pages 1441–1450, 2019.

[31] Hao Tan and Mohit Bansal. Vokenization: Improving language understanding with contextualized, visual-grounded supervision. *CoRR*, abs/2010.06775, 2020.

[32] Jiaxi Tang and Ke Wang. Personalized top-n sequential recommendation via convolutional sequence embedding. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining (WSDM 2018)*, pages 565–573, 2018.

[33] Yi Tay, Vinh Q. Tran, Mostafa Dehghani, Jianmo Ni, Dara Bahri, Harsh Mehta, Zhen Qin, Kai Hui, Zhe Zhao, Jai Gupta, Tal Schuster, William W. Cohen, and Donald Metzler. Transformer memory as a differentiable search index, 2022.

[34] Aäron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *CoRR*, abs/1807.03748, 2018.

[35] Shuyuan Xu, Wenyue Hua, and Yongfeng Zhang. Openp5: An open-source platform for developing, training, and evaluating llm-based recommender systems. *SIGIR*, 2024.

[36] Liu Yang, Fabian Paischer, Kaveh Hassani, Jiacheng Li, Shuai Shao, Zhang Gabriel Li, Yun He, Xue Feng, Nima Noorshams, Sem Park, Bo Long, Robert D. Nowak, Xiaoli Gao, and Hamid Eghbalzadeh. Unifying generative and dense retrieval for sequential recommendation. *Trans. Mach. Learn. Res.*, 2025, 2025.

[37] Neil Zeghidour, Alejandro Luebs, Ahmed Omran, Jan Skoglund, and Marco Tagliasacchi. Soundstream: An end-to-end neural audio codec. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 30:495–507, 2021.

[38] Kepu Zhang, Weijie Yu, Sunhao Dai, and Jun Xu. Citalaw: Enhancing llm with citations in legal domain, 2024.

[39] Jianan Zhao, Le Zhuo, Yikang Shen, Meng Qu, Kai Liu, Michael Bronstein, Zhaocheng Zhu, and Jian Tang. Graphtext: Graph reasoning in text space. In *Proceedings of the Thirty-Eighth Conference on Neural Information Processing Systems (NeurIPS 2024)*, 2024. Poster in the workshop *Adaptive Foundation Models: Evolving AI for Personalized and Efficient Learning*.

[40] Kun Zhou, Hui Wang, Wayne Xin Zhao, Yutao Zhu, Sirui Wang, Fuzheng Zhang, Zhongyuan Wang, and Ji-Rong Wen. S3-rec: Self-supervised learning for sequential recommendation with mutual information maximization. In *Proceedings of the 29th ACM International Conference on Information and Knowledge Management (CIKM 2020)*, pages 1893–1902, 2020.