D3: Dynamic Docid Decoding for Multi-Intent Generative Retrieval

Jaeyoung Kim[1]*, **Dohyeon Lee**[2]*, **Soona Hong**[2]*, **Seung-won Hwang**[1,2]†
Interdisciplinary Program in Artificial Intelligence, Seoul National University[1]
Computer Science and Engineering, Seoul National University[2]
{jae.young, waylight3, hongsoona, seungwonh}@snu.ac.kr

Abstract

Generative Retrieval (GR) maps queries to documents by generating discrete identifiers (DocIDs). However, offline DocID assignment and constrained decoding often prevent GR from capturing query-specific intent, especially when documents express multiple or unseen intents (i.e., intent misalignment). We introduce **D**ynamic **D**ocid **D**ecoding (D3), an inference-time mechanism that adaptively refines DocIDs through **delayed, query-informed identifier expansion**. D3 uses (a) *verification* to detect intent misalignment and (b) *dynamic decoding* to extend DocIDs with query-aligned tokens, even those absent from the pre-indexed vocabulary, enabling plug-and-play DocID expansion beyond the static vocabulary while adding minimal overhead. Experiments on NQ320k and MS-MARCO show that D3 consistently improves retrieval accuracy, especially on unseen and multi-intent documents, across various GR models, including a +2.4%p nDCG@10 gain on the state-of-the-art model.

1 Introduction

Generative Retrieval (GR) retrieves documents by generating their discrete identifiers (DocIDs), offering a lightweight alternative to dense retrieval (Lee et al., 2023a; Kuo et al., 2024; Zhang et al., 2024). By formulating retrieval as sequence generation, GR simplifies the information retrieval pipeline and reduces dependency on large vector indices (Sun et al., 2024).

However, current GR systems rely on DocIDs constructed offline, and decode only within a fixed prefix tree. This design creates a fundamental intent misalignment: the model may prefer tokens that better capture the query's intent but is restricted to those present in the pre-indexed DocID space (e.g., trie). As a result, GR often fails when documents express unseen or multiple intents not covered by the static DocIDs.

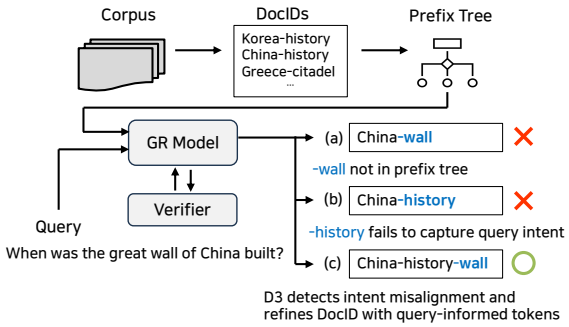

Figure 1: Comparison of standard GR with D3. (a) Query-aligned tokens (e.g., "wall") may not exist in the prefix tree, causing retrieval failure. (b) Constrained decoding forces selection among valid DocIDs, capturing only a coarse, high-level aspect of the query intent such as "China-history". (c) D3 detects intent misalignment through verification and dynamically extends DocIDs with query-aligned tokens, aligning retrieval with the user's intent.

Figure 1 illustrates this issue. (a) The offline DocID trie contains no branch encoding the intent-aligned combination "China–wall." (b) Constrained decoding thus settles for "China–history," the closest valid prefix, even if it does not match the query intent. Prior attempts to increase coverage such as assigning multiple DocIDs per document partially mitigate this problem (Bevilacqua et al., 2022; Li et al., 2023), but they incur high computational cost, risk identifier collisions, and still cannot anticipate all possible intents (Yuan et al., 2024).

To address these, we propose **D**ynamic **D**ocid **D**ecoding (D3), a delayed refinement at inference-time that combines **verification** with **dynamic decoding**. **Verification** compares constrained and unconstrained next-token distributions to detect when DocID prefix fails to capture the query intent. **Dynamic decoding** activates only when misalignment is detected, extending the prefix with

*Equal contribution
†Corresponding author

Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 5: Industry Track), pages 789–800
March 25-27, 2026

query-informed tokens, even if they lie outside the pre-indexed vocabulary. A lightweight document-aware lexical signal ensures the extended identifiers remain faithful to the underlying document, while reusing model logits keeps overhead minimal.

Experiments show that integrating D3 with existing GR models consistently improves performance while minimizing latency overhead. Notably, D3 yields average gains of +0.9%p on NQ320k and +3.2%p on MS-MARCO, with particularly strong gains on unseen and multi-intent settings. These results demonstrate that inference-time, query-adaptive refinement without retraining is a powerful and broadly applicable enhancement to GR systems.

2 Related Work

GR typically consists of two stages, indexing and decoding. We review prior works in each stage and discuss their limitations leading to misalignment.

Indexing in GR In the indexing stage, each document is mapped to a compact DocID intended to represent its semantics (Tay et al., 2022). Existing methods fall into two main categories. Multi-identifier indexing constructs multiple DocIDs per document to broaden intent coverage by enumerating diverse surface forms such as titles, substrings, or pseudo queries (Bevilacqua et al., 2022; Li et al., 2023, 2024a,b). Learnable DocID indexing learns a trainable DocID that captures document's dominant semantics and is optimized from the indexing objective (Lee et al., 2023b; Zhang et al., 2024; Zeng et al., 2024). Despite these advances, all of them still rely on offline indexing, which cannot incorporate query-informed signals at inference time.

Decoding in GR For a given query, the decoding stage generates sequences of discrete generation tokens from the model's vocabulary as DocIDs, typically using constrained beam search that restricts generation to the pre-indexed DocIDs. Recent improvements focus on reordering or reweighting these tokens within valid DocIDs (Zhang et al., 2024; Zeng et al., 2024), but none can generate tokens outside the indexed space. Thus, when the query requires tokens absent from pre-indexed trie, constrained decoding must settle for suboptimal alternatives, causing misalignment.

Our Distinction D3 improves accuracy through two components: **verification**, which detects when a DocID misaligns with a query intent, and **dynamic decoding**, which selectively extends DocIDs at inference time to better reflect query intent.

3 Proposed Method: D3

This section formally defines intent misalignment (§3.1) and introduces D3, which addresses it via verification (§3.2) and dynamic decoding (§3.3).

3.1 Motivation: Definition of Misalignment

In GR, the intent misalignment arises from the gap between the indexing and decoding stages, both of which prevent generating intent-aligned tokens even when they have high probability.

Offline DocID Indexing During indexing, each document d is assigned query-agnostic DocID z:

$$z = \underset{v}{\operatorname{argmax}}\, p(v|d), \tag{1}$$

optimized only from document content or training queries. Because this stage does not incorporate query-time intent, the resulting DocIDs often fail to represent unseen or multi-intent semantics that only become evident at inference.

Constrained Generation At inference time, a query q is mapped to DocID through constrained beam search:

$$p_\theta(z_{1:T}|q) = \prod_{t=1}^{T} p_\theta(z_t|z_{<t}, q) \quad (z_t \in V_t) \tag{2}$$

where V_t is the set of valid next tokens in prefix tree. If the model's preferred token lies outside this set, it is forced to choose a lower probability but permitted alternative.

Intent Misalignment We formalize misalignment by comparing the model's token choice **with** and **without** constraints, denoted as z_{w} and z_{wo}.

$$\begin{aligned} z_{\text{w}} &= \underset{z_t \in V_t}{\operatorname{argmax}}\, p_\theta(z_t|z_{<t}, q), \\ z_{\text{wo}} &= \underset{z_t \in V}{\operatorname{argmax}}\, p_\theta(z_t|z_{<t}, q), \end{aligned} \tag{3}$$

where V denotes entire token set. We define misalignment as $z_{\text{w}} \neq z_{\text{wo}}$, indicating that the model's true preference is blocked by the prefix tree.

As shown in Figure 2(a), a query about "the Great Wall of China" leads the unconstrained model to prefer "wall," but the trie lacks any China-wall branch. Constrained decoding instead outputs

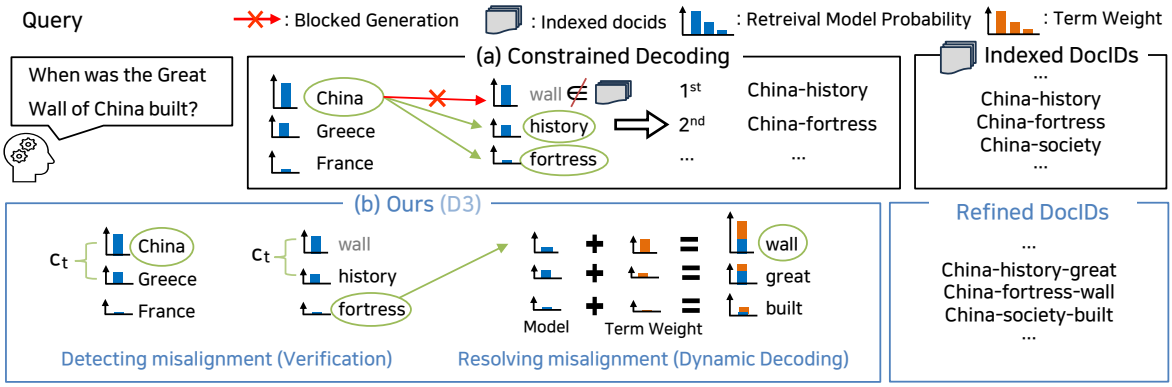

Figure 2: Comparison of (a) constrained decoding and (b) D3. In (a), constrained decoding restricts token selection to pre-indexed DocIDs in prefix tree. When a query requires tokens such as "wall" that do not exist along any valid path (no "China-wall" branch), the model is forced to choose suboptimal alternatives ("China-history"), leading to misalignment. In (b), D3 resolves this via two mechanisms: First, verification detects misalignment by computing the probability gap (c_t) between constrained and unconstrained distributions. When misalignment is detected, dynamic decoding extends DocID with query-aligned tokens ("wall," "great," "built"), weighted by both model probability and term weight, yielding extended DocID ("China-fortress-wall") that better capture the query intent.

Dataset	Rate of misalignment at t-step (%)						
	1^{st}	2^{nd}	3^{rd}	4^{th}	5^{th}	6^{th}	7^{th}
NQ320k	12.6	20.4	25.5	-	-	-	-
MS-MARCO	21.0	32.2	46.0	57.2	67.7	75.4	80.0

Table 1: Cumulative rate of queries where the generated tokens do not belong to the predefined index at each step. We use GLEN trained to produce DocIDs of length 3 on NQ320k and length 7 on MS-MARCO.

"history," yielding DocIDs such as China-history that miss the intended meaning. This example illustrates why conventional GR methods suffer from suboptimality by overlooking a misalignment.

Instead, we argue model should enforce alignment (i.e., $z_{\mathrm{w}} = z_{\mathrm{wo}}$) as queries and documents are not naturally aligned. To empirically show how the misalignment occurs in real-world scenario, we measured the cumulative rate of queries with this mismatch. As shown in Table 1, misalignment appears in 25.5% of NQ320k queries and 80% of MS-MARCO queries. These results support our motivation that offline indexing and constrained decoding limit the model's ability to capture query-aligned intent, necessitating dynamic refinement.

Figure 2(b) illustrates our goal to make the constrained choice z_{w} match the unconstrained preference z_{wo}. Simply expanding the DocID vocabulary is inefficient since most tokens (e.g., 74.5% in NQ320k) are already aligned. We thus propose a two-phase solution: (1) **detect** misalignment efficiently at inference time, and (2) **resolve** it by dynamically refining the DocID with query-relevant information. The following sections describe how D3 implements these steps.

3.2 Verification: Detect Misalignment

A straightforward way to detect intent misalignment is to directly check whether $z_{\mathrm{w}} \neq z_{\mathrm{wo}}$ holds. However, this binary rule is too brittle, as even minor or insignificant deviations are treated as misalignment. We instead relax this criterion by producing a continuous score, often called confidence (Wang and Zhou, 2024).

A naive formulation compares the probabilities of z_{w} and z_{wo},

$$c_t^{\text{naive}} = p_\theta(z_{\mathrm{w}}) - p_\theta(z_{\mathrm{wo}}), \tag{4}$$

but this score becomes unreliable when the next-token distribution is flat, making the difference uninformative. To obtain a more stable signal $\tilde{z}$, we adopt margin-based confidence that measures how much more probable the constrained choice is compared to the best alternative in the entire token set:

$$\begin{aligned} c_t &= p_\theta\left(z_{\mathrm{w}}\right) - p_\theta\left(\tilde{z}_{\mathrm{wo}}\right), \\ \tilde{z}_{\mathrm{wo}} &= \underset{\substack{z_t \in V, \\ z_t \neq z_{\mathrm{w}}}}{\operatorname{argmax}} p_\theta\left(z_t | z_{<t}, q\right). \end{aligned} \tag{5}$$

This margin naturally captures alignment. A high value indicates that z_{w} remains the model's global preference, while low values reveal that constrained decoding is blocking a more suitable token.

We compute an overall confidence $\bar{c}$ by averaging c_t across the prefix. If $\bar{c} \leq \alpha$, we deem the prefix insufficiently aligned and activate dynamic decoding. Importantly, this verification step is efficient, as it reuses the probability distributions

already computed during beam search. Thus, it provides a reliable and low-cost mechanism for identifying when dynamic refinement is truly needed.

3.3 Dynamic Decoding: Resolve Misalignment

When verification detects a significant misalignment ($\bar{c} < \alpha$), D3 triggers dynamic decoding, extending the DocID beyond its predefined length T. The goal is to add only the minimal set of query-informed tokens needed to restore alignment, while preserving relevance to the underlying document. A naive approach would search over the entire tokens V, but this is inefficient and likely to introduce noise. Instead, D3 employs a compact, interpretable candidate set and applies a lightweight document-aware scoring mechanism.

Query-Informed Vocabulary Most diagnostic intent-bearing tokens are explicitly present in the query (Saha Roy et al., 2015). Thus, rather than exploring all of V, we construct a query-informed vocabulary:

$$V_q = Q \setminus \{z_{T+1}, \ldots, z_{t-1}\}, \tag{6}$$

where Q is the set of unique query tokens, and previously generated ones are excluded. This drastically reduces the candidate space and ensures that dynamic decoding focuses on terms most likely to capture the missing intent[1].

Document-Aware Token Scoring Selecting the highest probability token from V_q alone may produce tokens irrelevant to the actual document. To maintain document fidelity, we combine the model's preference with lightweight lexical signal:

$$\text{where score}_t(z) = p_\theta(z|z_{<t}, q) \cdot s(z, d), \tag{7}$$

where $s(z, d)$ is a document-aware term weight[2]. The next token is selected as:

$$\hat{z}_w = \underset{z_t \in V_q}{\text{argmax}} \ \text{score}_t(z_t) \tag{8}$$

This scoring serves as a soft conjunction between two complementary signals. The model probability $p_\theta(z|z_{<t}, q)$ captures semantic relevance to the query, while the term weight $s(z, d)$ reflects lexical faithfulness to the underlying document. This design discourages spurious query terms that are semantically plausible but absent from the document, as well as document terms that are irrelevant to the query. Overall process of D3 are described in Appendix A.1.

[1] An ablation of V_q is presented in Appendix A.6.

[2] We use BM25, and other methods are explored in Appendix A.7.

4 Experiments

4.1 Experimental Setting

Dataset We evaluate D3 on two widely used datasets across different retrieval scenarios. (1) NQ320k (Kwiatkowski et al., 2019) is used to evaluate retrieval performance in a knowledge-intensive QA setting. Following prior works (Lee et al., 2023b; Sun et al., 2024), we split test queries into seen and unseen based on whether their annotated target documents appear as ground truth in the training queries. (2) MS-MARCO Passage (Bajaj et al., 2016) is a large-scale passage retrieval dataset designed for real-world applications. We evaluate on TREC DL 2019 (Craswell et al., 2019) and 2020 (Craswell et al., 2021), two standard benchmarks for ranking quality at scale. We use standard ranking metrics (MRR, Recall, nDCG), with detailed dataset statistics and metric definitions provided in Appendix A.2 and A.3.

Baselines To ensure an architecture-agnostic and comprehensive comparison, we evaluate D3 on both learnable DocID models and multi-identifier models, covering the major design choices in GR. For NQ320k, we include TSGen (Zhang et al., 2024) and GLEN (Lee et al., 2023b), two of the strongest models in the learnable DocID setting. To further evaluate D3 under multiple DocIDs setting, we add SEAL (Bevilacqua et al., 2022) and MINDER (Li et al., 2023), both of which explicitly model diverse query intents. For MS-MARCO Passage, we use PAG (Zeng et al., 2024), a state-of-the-art model, along with LTRGR (Li et al., 2024a) and DGR (Li et al., 2024b) for broader coverage.

Implementation Details See Appendix A.4

4.2 Effectiveness Analysis

We show D3 consistently improves performance as a plug-and-play component across diverse GR models, indicating its architecture-agnostic nature.

Effectiveness on Knowledge-Intensive Dataset (Table 2) NQ320k contains knowledge-intensive queries and exhibits a well-known performance gap between seen and unseen documents (Sun et al., 2024; Zhang et al., 2024). As shown in Table 2, applying D3 consistently increases confidence scores

Model	Full (6,330)				Seen (4,911)				Unseen (1,419)			
	R@1	R@10	MRR@100	Conf.	R@1	R@10	MRR@100	Conf.	R@1	R@10	MRR@100	Conf.
BM25	29.4	60.1	39.9	-	28.8	59.7	39.2	-	31.4	61.6	42.1	-
SEAL	56.0	81.2	65.3	0.531	64.3	86.8	72.9	0.592	27.5	61.7	39.1	0.317
+ D3	**57.6** (+1.6)	**83.4** (+2.2)	**67.2** (+1.9)	**0.655** (+0.124)	**64.9** (+0.6)	**87.9** (+1.1)	**73.7** (+0.8)	**0.668** (+0.076)	**32.4** (+4.9)	**67.9** (+6.2)	**44.5** (+5.4)	**0.623** (+0.306)
MINDER	62.4	84.4	70.6	0.564	69.3	88.3	76.4	0.621	38.4	71.0	50.5	0.368
+ D3	**63.3** (+0.9)	**85.7** (+1.3)	**71.6** (+1.0)	**0.665** (+0.101)	**69.8** (+0.5)	**89.0** (+0.7)	**77.0** (+0.6)	**0.677** (+0.056)	**40.9** (+2.5)	**74.5** (+3.5)	**53.1** (+2.6)	**0.623** (+0.255)
GLEN	69.0	85.6	75.1	0.732	72.4	88.5	78.3	0.756	57.4	75.6	64.0	0.651
+ D3	**69.6** (+0.6)	**86.7** (+1.1)	**75.8** (+0.7)	**0.918** (+0.186)	**72.9** (+0.5)	**89.4** (+0.9)	**78.9** (+0.6)	**0.798** (+0.042)	**58.1** (+0.7)	**77.7** (+2.1)	**65.3** (+1.3)	**0.708** (+0.057)
TSGen	70.4	88.0	77.1	0.950	71.1	88.4	77.8	0.951	67.7	**86.5**	74.8	0.947
+ D3	**70.8** (+0.4)	**88.1** (+0.1)	**77.4** (+0.3)	**0.960** (+0.010)	**71.4** (+0.3)	**88.6** (+0.2)	**78.0** (+0.2)	**0.959** (+0.008)	**68.5** (+0.8)	**86.5** (+0.0)	**75.3** (+0.5)	**0.963** (+0.016)

Table 2: Performance comparison for the proposed method and baseline models on NQ320k. The best performance for each GR model is marked bold. R and Conf. indicate Recall and average confidence score, respectively.

Model	TREC DL 2019			TREC DL 2020		
	nDCG	R	Conf.	nDCG	R	Conf.
BM25	50.6	12.9	-	48.0	16.4	-
GLEN	47.5	10.2	0.366	46.2	15.9	0.376
+ D3	**51.5** (+4.0)	**13.2** (+3.0)	**0.560** (+0.194)	**51.6** (+5.4)	**16.2** (+0.3)	**0.464** (+0.088)
LTRGR	59.8	15.2	0.093	55.5	18.2	0.096
+ D3	**62.6** (+2.8)	**16.1** (+0.9)	**0.709** (+0.616)	**58.5** (+3.0)	**19.7** (+1.5)	**0.740** (+0.644)
DGR	61.2	15.1	0.096	57.7	20.1	0.091
+ D3	**64.7** (+3.5)	**16.3** (+1.2)	**0.785** (+0.689)	**61.2** (+3.7)	**20.5** (+0.4)	**0.873** (+0.782)
PAG	70.5	26.7	-0.091	70.0	23.6	-0.082
+ D3	**72.9** (+2.4)	**28.0** (+1.3)	**0.548** (+0.639)	**70.4** (+0.4)	**23.8** (+0.2)	**0.555** (+0.637)

Table 3: Performance comparison of D3 and baselines on MS-MARCO. All metrics are reported at @10, and the best result for each model is shown in bold.

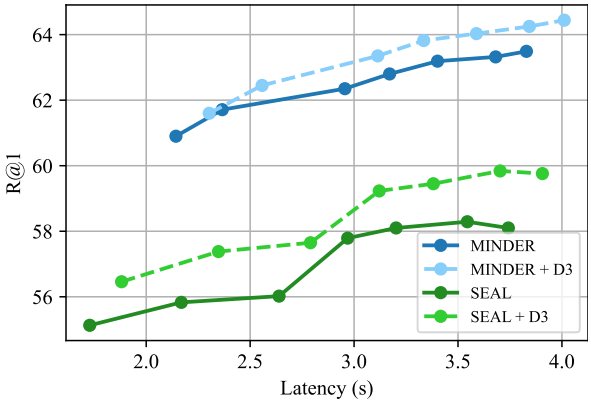

Figure 3: R@1 versus inference latency comparing baselines (solid lines) and the same models with D3 (dotted lines) on NQ320k.

across all evaluated GR models. Specifically, this higher confidence, indicating a better alignment between the generated DocIDs and query intents, translates into substantial gains on unseen documents. For instance, TSGen, the strongest baseline, achieves a +0.8%p gain in Recall@1 on the unseen split of NQ320k when combined with D3.

Effectiveness on Large-Scale Dataset (Table 3) MS-MARCO Passage poses a different challenge: its large corpus (8.8M passages) suffers from more frequent intent misalignment, resulting in lower baseline confidence. As shown in Table 3, D3 effectively boosts confidence, which in turn drives retrieval improvements. Notably, D3 also improves PAG, which combines lexical and numerical DocIDs, showing that our method is compatible with hybrid identifier design. For example, on TREC DL 2019, D3 elevates PAG's confidence from -0.091 to 0.548 (+0.639), yielding a +2.4%p gain in nDCG.

These findings confirm that by systematically mitigating intent misalignment to increase confidence, D3 remains effective even for large-scale retrieval.

Qualitative Behavior Beyond these quantitative gains, D3 also produces qualitatively more intent-aligned DocIDs. Case studies in Appendix A.5 show that D3 selectively adds only the minimal set of essential tokens such as "congress" or "power" depending on query needs, when predefined DocIDs fail to encode fine-grained query intent. These examples further validate that D3 adapts DocIDs in a targeted manner without over-refinement.

4.3 Efficiency Analysis

To increase intent coverage, SEAL uses all substrings of a document as DocIDs, and MINDER further incorporates pseudo queries as additional DocIDs, sacrificing inference latency. To evaluate efficiency, we compare D3 against these approaches by manipulating the number of DocIDs. Low-latency settings are simulated by truncating documents to

Model	Rate	R@1	Conf.
SEAL	91.1%	55.9	0.508
+ D3		**57.6**	**0.639**
MINDER	94.5%	61.9	0.530
+ D3		**62.9**	**0.640**
GLEN	36.2%	68.9	0.303
+ D3		**69.7**	**0.814**
TSGen	6.5%	22.6	0.712
+ D3		**28.2**	**0.864**

Table 4: Analysis of the NQ320k queries that trigger D3 during decoding.

Model	Easy-Intent		Hard-Intent	
	R@1	Conf.	R@1	Conf.
SEAL	**71.7**	0.651	40.2	0.410
+ D3	71.6	**0.676**	**43.5**	**0.634**
MINDER	**76.6**	0.667	48.0	0.461
+ D3	**76.6**	**0.682**	**49.9**	**0.647**
GLEN	75.8	0.790	62.1	0.675
+ D3	**76.2**	**0.935**	**62.8**	**0.901**
TSGen	77.2	0.966	63.5	0.935
+ D3	**77.4**	**0.970**	**64.1**	**0.950**

Table 5: Comparison between Easy-Intent and Hard-Intent queries on NQ320k.

reduce substring-based DocIDs, while high-latency settings use DocT5Query expansion (Nogueira et al., 2019) to enlarge the DocID set.

Figure 3 shows Recall@1 against inference latency for all scenario. Across all latency points, applying D3 consistently outperforms the corresponding SEAL and MINDER baselines. This shows that dynamic DocID refinement at inference is substantially more efficient than offline DocID expansion that attempt to increase intent coverage by expanding the DocID set. Furthermore, unlike DocT5Query-based expansion, D3 introduces no additional indexing overhead, providing both stronger performance and lower latency. These properties make D3 particularly suitable for large-scale, continuously evolving corpora where index updates are costly in production GR systems.

An ablation study in Appendix A.6 further highlights the role of verification: removing it activates dynamic decoding for nearly all queries, increasing latency without performance gains. This confirms that verification is essential for keeping D3 efficient by activating only when misalignment is detected.

4.4 Deeper Analysis

D3 selectively refines DocIDs (Table 4). To understand how well D3 identifies misalignment, we analyze the subset of queries flagged by verification. Table 4 shows that weaker models (SEAL, MINDER) trigger refinement for nearly all queries (91.1%, 94.5%), whereas stronger models (GLEN, TSGen) refine only a small fraction (36.2%, 6.5%). This selective refinement demonstrates that D3 avoids unnecessary modifications and focuses on queries with poor intent alignment. Furthermore, applying D3 to these misaligned queries yields substantial gains in both confidence score and performance. For instance, in TSGen, the confidence score increases from 0.712 to 0.864, and Recall@1 rises from 22.6 to 28.2 after refinement. These results show that D3 improves retrieval not by over-generating, but by targeted corrections on the exact queries suffering from intent misalignment.

Model	ROUGE-L	LLM Eval
GLEN	25.4	52.5
+ D3	**26.8**	**54.5**
LTRGR	27.3	56.7
+ D3	**27.7**	**58.0**
DGR	27.2	57.1
+ D3	**27.9**	**58.4**
PAG	28.2	58.7
+ D3	**28.4**	**59.7**

Table 6: Question Answering performance on MS-MARCO with Llama-3.1-8B-Instruct.

D3 resolves multi-intent problem (Table 5). Real-world documents often express multiple intents, but only a subset of these intents is typically observed during training. Similar to prior work (Zhan et al., 2022), we split test queries into Easy-Intent and Hard-Intent subsets based on how well their documents' intents were covered in the training data (See Appendix A.8). Table 5 shows that Hard-Intent queries indeed exhibit lower baseline confidence and degraded performance. Importantly, D3 yields substantially larger improvements on Hard-Intent queries across all models, indicating its ability to dynamically recover query-aligned intent even when the corresponding document intent is rarely observed during training.

D3 improves downstream task performance (Table 6). To examine whether resolving intent misalignment benefits end-to-end applications, we evaluate D3 in a RAG-style question answering setup. As shown in Table 6, using the top-10 documents retrieved with D3 consistently improves QA performance across all GR models. In particular,

D3 yields higher ROUGE-L scores and achieves at least +1.0%p gain on LLM Eval, demonstrating that more intent-aligned DocIDs lead to higher-quality retrieved documents and, consequently, better downstream reasoning. These results confirm that the advantages of D3 extend beyond retrieval metrics, enhancing the overall effectiveness of real-world IR pipelines.

5 Conclusion

In this paper, we introduced D3, an inference-time solution for addressing intent misalignment in GR. This approach fundamentally shifts the retrieval paradigm from relying on static, query-agnostic identifiers to creating dynamic, query-aware ones. Experiments on NQ320k and MS-MARCO show that D3 yields significant gains, especially for documents with diverse intents, highlighting the potential of inference-time adaptive retrieval in large-scale systems.

6 Limitations

D3 delivers strong performance and adaptability, but it also has limitations. First, it uses dynamic decoding, which, while efficient, may cause slight latency compared to fully static methods when intent extraction is triggered frequently. Second, while D3 does not modify the underlying index, it also does not explicitly optimize for newly added documents. Nevertheless, D3 remains fully compatible with existing GR pipelines and requires neither retraining nor re-indexing when documents are added or updated.

Acknowledgements

This work was partly supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) [NO.RS-2021-II211343, Artificial Intelligence Graduate School Program (Seoul National University)] and the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIT) (No. RS-2024-00414981).

References

AI@Meta. 2024. Llama 3 model card.

Payal Bajaj, Daniel Campos, Nick Craswell, Li Deng, Jianfeng Gao, Xiaodong Liu, Rangan Majumder, Andrew McNamara, Bhaskar Mitra, Tri Nguyen, et al. 2016. MS MARCO: A human generated machine reading comprehension dataset. *arXiv preprint arXiv:1611.09268*.

Michele Bevilacqua, Giuseppe Ottaviano, Patrick Lewis, Scott Yih, Sebastian Riedel, and Fabio Petroni. 2022. Autoregressive search engines: Generating substrings as document identifiers. *Advances in Neural Information Processing Systems*, 35:31668–31683.

Nick Craswell, Bhaskar Mitra, Emine Yilmaz, and Daniel Campos. 2019. Overview of the trec 2019 deep learning track. In *TREC*.

Nick Craswell, Bhaskar Mitra, Emine Yilmaz, Daniel Fernando Campos, and Ellen M. Voorhees. 2021. Overview of the trec 2020 deep learning track. volume abs/2102.07662.

Thibault Formal, Carlos Lassance, Benjamin Piwowarski, and Stéphane Clinchant. 2022. From distillation to hard negative sampling: Making sparse neural ir models more effective.

Tzu-Lin Kuo, Tzu-Wei Chiu, Tzung-Sheng Lin, Sheng-Yang Wu, Chao-Wei Huang, and Yun-Nung Chen. 2024. A survey of generative information retrieval. *arXiv preprint arXiv:2406.01197*.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, et al. 2019. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466.

Sunkyung Lee, Minjin Choi, and Jongwuk Lee. 2023a. GLEN: Generative retrieval via lexical index learning. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 7693–7704, Singapore. Association for Computational Linguistics.

Sunkyung Lee, Minjin Choi, and Jongwuk Lee. 2023b. Glen: Generative retrieval via lexical index learning. *arXiv preprint arXiv:2311.03057*.

Yongqi Li, Nan Yang, Liang Wang, Furu Wei, and Wenjie Li. 2023. Multiview identifiers enhanced generative retrieval. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6636–6648.

Yongqi Li, Nan Yang, Liang Wang, Furu Wei, and Wenjie Li. 2024a. Learning to rank in generative retrieval. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 8716–8723.

Yongqi Li, Zhen Zhang, Wenjie Wang, Liqiang Nie, Wenjie Li, and Tat-Seng Chua. 2024b. Distillation enhanced generative retrieval. *arXiv preprint arXiv:2402.10769*.

Chin-Yew Lin and Franz Josef Och. 2004. Automatic evaluation of machine translation quality using longest common subsequence and skip-bigram statistics. In *Proceedings of the 42nd annual meeting of the association for computational linguistics (ACL-04)*, pages 605–612.

Jimmy Lin, Xueguang Ma, Sheng-Chieh Lin, Jheng-Hong Yang, Ronak Pradeep, and Rodrigo Nogueira. 2021. Pyserini: A Python toolkit for reproducible information retrieval research with sparse and dense representations. In *Proceedings of the 44th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2021)*, pages 2356–2362.

Rodrigo Nogueira, Wei Yang, Jimmy Lin, and Kyunghyun Cho. 2019. Document expansion by query prediction. *arXiv preprint arXiv:1904.08375*.

Rishiraj Saha Roy, Rahul Katare, Niloy Ganguly, Srivatsan Laxman, and Monojit Choudhury. 2015. Discovering and understanding word level user intent in web search queries. *Journal of Web Semantics*, 30:22–38. Semantic Search.

Weiwei Sun, Lingyong Yan, Zheng Chen, Shuaiqiang Wang, Haichao Zhu, Pengjie Ren, Zhumin Chen, Dawei Yin, Maarten Rijke, and Zhaochun Ren. 2024. Learning to tokenize for generative retrieval. *Advances in Neural Information Processing Systems*, 36.

Yi Tay, Vinh Tran, Mostafa Dehghani, Jianmo Ni, Dara Bahri, Harsh Mehta, Zhen Qin, Kai Hui, Zhe Zhao, Jai Gupta, et al. 2022. Transformer memory as a differentiable search index. *Advances in Neural Information Processing Systems*, 35:21831–21843.

Xuezhi Wang and Denny Zhou. 2024. Chain-of-thought reasoning without prompting, 2024. *URL https://arxiv. org/abs/2402.10200*.

Xiao Yang, Kai Sun, Hao Xin, Yushi Sun, Nikita Bhalla, Xiangsen Chen, Sajal Choudhary, Rongze Gui, Ziran Jiang, Ziyu Jiang, et al. 2024. Crag-comprehensive rag benchmark. *Advances in Neural Information Processing Systems*, 37:10470–10490.

Peiwen Yuan, Xinglin Wang, Shaoxiong Feng, Boyuan Pan, Yiwei Li, Heda Wang, Xupeng Miao, and Kan Li. 2024. Generative dense retrieval: Memory can be a burden. *arXiv preprint arXiv:2401.10487*.

Hansi Zeng, Chen Luo, and Hamed Zamani. 2024. Planning ahead in generative retrieval: Guiding autoregressive generation through simultaneous decoding. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 469–480.

Jingtao Zhan, Xiaohui Xie, Jiaxin Mao, Yiqun Liu, Jiafeng Guo, Min Zhang, and Shaoping Ma. 2022. Evaluating interpolation and extrapolation performance of neural retrieval models. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, pages 2486–2496.

Peitian Zhang, Zheng Liu, Yujia Zhou, Zhicheng Dou, Fangchao Liu, and Zhao Cao. 2024. Generative retrieval via term set generation. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 458–468.

A Appendix

A.1 Overall Process of D3

Algorithm 1 outlines how verification and dynamic decoding interact during inference. The model first generates a predefined DocID prefix of length T, and computes the average confidence score $\bar{c}_T$ (line 4). If $\bar{c}_T > \alpha$, the prefix is deemed aligned and returned directly (lines 5–6). Otherwise, D3 enters dynamic decoding. In this phase, the prefix is extended one token at a time, and a new confidence score $\bar{c}_t$ is computed to evaluate whether the refinement improves alignment. Dynamic decoding terminates when one of the following holds: (i) Alignment restored (line 11): $\bar{c}_t > \alpha$, meaning the extended prefix now matches the model's unconstrained preference. (ii) Quality degradation (line 14): The confidence drops sharply, $\bar{c}_{t-1} - \bar{c}_t > \beta$, where β is a stability threshold preventing over-refinement; the algorithm then returns the prefix at step $t-1$. (iii) Maximum length reached (line 8): The sequence length reaches $2T$, ensuring bounded computation. Overall, verification provides an efficient mechanism to determine when refinement is needed, while dynamic decoding selectively resolves misalignment with minimal overhead. Details of the hyperparameters α and β are provided in Appendix A.4.

A.2 Dataset Details

NQ320k (Kwiatkowski et al., 2019) consists of 109k documents, 320k training queries, and 7,830 test queries. Following prior work (Lee et al., 2023b; Sun et al., 2024), we split the test queries into the seen (6,075) and unseen (1,755) subsets depending on whether their annotated documents appear as ground truth in the training set. MS-MARCO Passage (Bajaj et al., 2016) includes 500k training queries and 6,980 development queries. Based on this dataset, TREC DL 2019 (Craswell et al., 2019) and TREC DL 2020 (Craswell et al., 2021) provide evaluation benchmarks containing 43 and 54 queries, respectively. For all experiments, we exclude the queries for hyperparameter search, as described in Appendix A.4.

α	Model (Recall@1)			
	SEAL	MINDER	GLEN	TSGen
0.0	57.80	63.60	70.00	72.73
0.1	57.93	63.53	70.00	72.73
0.2	58.00	63.60	70.06	72.73
0.3	58.13	63.60	70.06	72.73
0.4	58.27	63.73	70.06	72.73
0.5	58.20	64.00	70.20	72.73
0.6	58.47	64.07	70.20	72.73
0.7	58.40	64.00	70.26	72.80
0.8	58.47	64.00	**70.53**	**72.87**
0.9	**58.53**	**64.07**	70.53	72.80
1.0	58.40	64.07	70.53	72.80
w/o D3	57.27	63.40	69.94	72.50

(a) Searching α (β=0.0)

β	Model (Recall@1)			
	SEAL	MINDER	GLEN	TSGen
0.0	58.53	64.07	70.53	72.87
0.1	**59.80**	**64.67**	70.60	72.93
0.2	59.60	64.40	70.80	**73.00**
0.3	59.53	64.40	71.00	73.00
0.4	59.00	64.33	71.00	73.00
0.5	58.80	64.20	71.13	73.00
0.6	58.93	64.40	71.13	73.00
0.7	58.87	64.33	71.13	73.00
0.8	58.87	64.33	71.13	73.00
0.9	58.87	64.33	71.13	73.00
1.0	58.87	64.33	**71.20**	73.00
w/o D3	57.27	63.40	69.94	72.50

(b) Searching β (best α per model)

Table 7: Recall@1 performance on validation queries of NQ320k to select hyperparameter α and β.

α	Model (MRR@10)			
	GLEN	LTRGR	DGR	PAG
0.0	20.43	25.50	27.60	40.20
0.1	20.56	25.90	27.60	39.90
0.2	20.70	25.90	27.70	39.90
0.3	20.97	26.00	27.80	39.80
0.4	21.15	26.10	27.80	**40.50**
0.5	21.37	**26.30**	28.00	40.50
0.6	21.62	26.00	27.90	40.20
0.7	**21.75**	25.90	28.00	40.50
0.8	21.55	25.90	**28.10**	40.50
0.9	21.75	25.80	28.10	40.30
w/o D3	20.43	25.40	27.50	40.10

(a) Searching α (β=0.0)

β	Model (MRR@10)			
	GLEN	LTRGR	DGR	PAG
0.0	21.45	27.00	28.30	38.70
0.1	22.44	27.50	28.10	38.70
0.2	22.90	27.60	28.20	38.70
0.3	23.38	27.50	**28.40**	38.70
0.4	22.83	27.70	28.30	**38.80**
0.5	22.87	27.80	28.20	38.80
0.6	23.58	**27.90**	28.30	38.80
0.7	**23.88**	27.90	28.30	38.80
0.8	23.72	27.90	28.30	38.80
0.9	23.54	27.90	28.30	38.80
w/o D3	20.26	26.40	27.40	37.90

(b) Searching β (best α per model)

Table 8: MRR@10 performance on validation queries of MS-MARCO to select hyperparameter α and β.

A.3 Metric Details

We evaluate retrieval performance using MRR, Recall, and nDCG. MRR measures ranking quality by assessing the rank of the first relevant document. Recall measures the proportion of relevant documents retrieved. nDCG evaluates ranking quality with graded relevance scores. All metrics are calculated within the top-k results. These metrics align with prior GR benchmarks, ensuring fair comparison with baseline models.

To evaluate D3 on downstream question answering task in Section 4.4, we use ROUGE-L and LLM Eval. ROUGE-L (Lin and Och, 2004) compares predicted answers to ground truth answers based on lexical overlap. For LLM Eval, we use Llama-3.3-70B-Instruct (AI@Meta, 2024), following prior work (Yang et al., 2024) which uses an LLM as the judge.

A.4 Implementation Details

For all baselines, we use the official checkpoints released by the authors. Since SEAL and MINDER do not provide NQ320k checkpoints, we reproduced their models using the hyperparameters reported in their papers. For the term-weighting function in D3, we use `pyserini` (Lin et al., 2021) to compute BM25 scores with default settings.

D3 introduces two hyperparameters: the verification threshold α and the degradation threshold β. To apply D3 to each baseline fairly, we perform a lightweight hyperparameter search on a small held-out subset (1,500 queries from NQ320k and 1,000 queries from the MS-MARCO development set), which is excluded from all other experiments. We

Relevant document	Query	Refined DocID
United States Congress 535 voting members 100 senators 435 representatives 6 non-voting members Senate political groups Republican (51) Democratic (47) Independent (2) (caucusing with Democrats) House of Representatives political groups Republican (237) Democratic (193) Vacant (5) Elections Senate last election November 8, 2016 ...	how many members in the senate are democratic	representatives-democratic-election
	to which groups are members of congress responsible	representatives-democratic-election-congress
	who has the most real power in the house of representatives	representatives-democratic-election-power-real-has

Table 9: Examples of Refined DocIDs on NQ320k. These queries share the same relevant document but differ in their query intent. Blue indicates query-agnostic tokens, and Red denotes the query-aligned tokens to refine DocID. For clarity, subword tokens were combined and displayed as whole words.

Algorithm 1 Overall Process of D3

Input: Query q, threshold α, degradation threshold β
Output: Final DocID $z_{1:T'}$
Initialize $t \leftarrow T+1$ ▷ Start after predefined length
Compute initial $\bar{c}_T$ from predefined prefix
if $\bar{c}_T \geq \alpha$ **then**
  return predefined DocID $z_{1:T}$ ▷ No refinement
end if
while $t \leq 2T$ **do** ▷ Maximum length constraint
  Generate next token z_t using dynamic decoding
  Compute confidence score c_t and average $\bar{c}_t$
  if $\bar{c}_t > \alpha$ **then** ▷ Condition 1
    return $z_{1:t}$
  end if
  if $\bar{c}_{t-1} - \bar{c}_t > \beta$ **then** ▷ Condition 2
    return $z_{1:t-1}$ ▷ Revert to previous step
  end if
  $t \leftarrow t+1$
end while
return $z_{1:2T}$ ▷ Return at maximum length

Model	Method	R@1
SEAL	+D3	**57.6**
	w/o verification α	56.9
	w/o query-informed vocabulary V_q	54.4
	w/o term weight $s(z,d)$	56.6
MINDER	+D3	**63.3**
	w/o verification α	60.3
	w/o query-informed vocabulary V_q	58.9
	w/o term weight $s(z,d)$	62.6
GLEN	+D3	**69.6**
	w/o verification α	67.8
	w/o query-informed vocabulary V_q	69.3
	w/o term weight $s(z,d)$	68.7
TSGen	+D3	**70.8**
	w/o verification α	70.7
	w/o query-informed vocabulary V_q	70.5
	w/o term weight $s(z,d)$	70.4

Table 10: Ablation study for each module in D3 on NQ320k.

adopt a sequential greedy search (first selecting α with β fixed to 0.0, and then selecting β with α fixed) as shown in Table 7 and Table 8. If multiple candidates yield the same validation performance, we choose the smaller value. Because GR models generate deterministic top-k tokens during beam search, all results are fully reproducible.

Importantly, we observe that D3 consistently improves performance over most baselines across a wide range of α and β settings. The gains are robust to hyperparameter choices, indicating that the performance improvements stem from the mechanism of dynamic DocID refinement itself rather than from hyperparameter tuning.

A.5 Case Study

To qualitatively illustrate how dynamic decoding refines DocIDs based on query intent, we present a case study in Table 9 using three queries that share the same relevant document. The first query targets general information from the document, that is well-covered by the predefined DocID, requiring no refinement. The second query seeks more specific details about congress, prompting the model to append the token "congress" to the original DocID. The third query demands highly specific information—identifying who holds the greatest power in the House of Representatives—leading to the addition of the query-informed tokens "power", "real", and "has". These examples demonstrate that our approach adaptively refines DocIDs with only the necessary query-aligned tokens, depending on how well the predefined DocID already captures the query intent.

A.6 Ablation Study

We conduct an ablation study to examine the contribution of each module in D3, with results summarized in Table 10 for retrieval performance.

Model	Method	Latency	R@1
SEAL	+D3	**1.00x**	**57.6**
	w/o verification α	1.05x	56.9
MINDER	+D3	**1.00x**	**63.3**
	w/o verification α	1.33x	60.3
GLEN	+D3	**1.00x**	**69.6**
	w/o verification α	1.39x	67.8
TSGen	+D3	**1.00x**	**70.8**
	w/o verification α	1.26x	70.7

Table 11: Ablation study for the verification in D3 on NQ320k. Latency indicates the relative time required to retrieve documents for a query. The best results are marked in bold.

The verification module is designed to identify queries whose predefined DocIDs are misaligned, ensuring that dynamic decoding is applied selectively. When verification is removed, i.e., w/o verification α, dynamic decoding is applied to all queries regardless of whether refinement is needed. This results in only marginal gains or even degrades performance compared to the baselines, while also incurring substantial inference latency, as shown in Table 11. These findings confirm that verification effectively detects genuinely misaligned queries and directs refinement to cases where it is most beneficial.

The query-informed vocabulary guides the model to select query-aligned tokens during decoding. Removing this restriction, i.e., w/o query-informed vocabulary V_q, allows the model to consider the entire token set V for next token prediction, which consistently reduces performance across all models. This demonstrates that narrowing candidate set to query-aligned tokens is critical for capturing query intent, even for models trained to handle larger candidate sets.

The term weighting module ensures that selected tokens remain faithful to the content of the document. Omitting term weights in Eq. (7), i.e., w/o term weight $s(z, d)$, consistently lowers performance, highlighting the importance of weighting tokens according to document relevance to maintain content fidelity.

Overall, these ablation results indicate that each module in D3 serves a distinct purpose and contributes meaningfully to its overall effectiveness. Their combined operation is essential for achieving the observed improvements in retrieval accuracy.

Method	Model			
	SEAL	MINDER	GLEN	TSGen
Baseline	56.0	62.4	69.0	70.4
+ D3 (BBoW)	56.6	62.6	69.2	70.6
+ D3 (BM25)	57.6	63.3	**69.6**	**70.8**
+ D3 (SPLADE)	**59.0**	**64.0**	69.1	**70.8**

Table 12: R@1 performance on NQ320k for each document term weight function. In BBoW (Binary Bag-of-Words) setting, term weights are either 1 or 0 depending on whether the document contains the term.

A.7 Generalize to Diverse Term Weights

Table 12 demonstrates that D3 consistently improves performance across all term weighting strategies, including Binary Bag-of-Words (BBoW), BM25, and SPLADE (Formal et al., 2022). Even with BBow, applying D3 yields gains of +0.3%p on average across models. Furthermore, BM25 and SPLADE provide slightly higher improvements of +0.8%p and +1.2%p, respectively, indicating that the performance boost is not solely due to sophisticated term weighting. Importantly, D3 generalizes well across all term weight schemes, and even the simplest BBoW produces meaningful gains, highlighting the robustness and broad applicability of D3 across diverse term scoring approaches.

A.8 Intent Coverage and Multi-Intent Partition

Many documents exhibit multiple possible intents, only some of which appear during training. To quantify this, we define a document's intent coverage as the difference between the number of training queries for which the document is relevant and the number of test queries for which it is relevant. A higher coverage value indicates that the document's intents were frequently observed during training, while lower values indicate rarely seen or unseen intents. Similar to prior work (Zhan et al., 2022), we use this metric to divide test queries into two equally sized subsets: Easy-Intent (top 50% coverage) and Hard-Intent (bottom 50% coverage). Hard-Intent queries represent challenging cases where static DocIDs often misalign with query-aligned intents.

As shown in Table 5, Hard-Intent queries exhibit noticeably lower baseline confidence and degraded retrieval performance. D3 delivers significantly larger improvements on this subset because its dynamic refinement mechanism introduces intent-

bearing tokens that are missing from static DocIDs, thereby resolving misalignment caused by unseen or underrepresented intents during training.