Query-Aware Token Budgeting for Efficient Late-Interaction Visual Document Retrieval

PS Rishi*, Rajeev Ranjan Dwivedi*†, Vinod Kumar Kurmi
Indian Institute of Science Education and Research Bhopal (IISER Bhopal), India
rishi.sivakumar10@gmail.com, rajeevias95@gmail.com, vinodkk@iiserb.ac.in
*Equal contribution. †Corresponding author.

***Abstract*—Late-interaction visual document retrievers preserve fine-grained page evidence by storing many token embeddings per page, but the resulting storage and query-time interaction cost makes large-scale deployment expensive. A natural remedy is to pool document tokens before indexing, yet static pooling must decide which visual evidence to preserve before the query is known. This paper studies an alternative: use a heavily compressed index only for candidate generation, then perform query-aware token budgeting on the original token set for shortlisted pages. We formulate stage-two token selection as a budgeted MaxSim coverage problem, show that a clipped version is monotone submodular, and compare coverage-only, cluster-guided, token-wise, and marginal-gain selection policies. On ten ViDoRe tasks with ColModernVBERT, direct static pooling drops macro normalized discounted cumulative gain at rank five from 0.6309 without compression to 0.4738 at a thirty-two-fold pool factor. Under the same candidate-generation regime and a pool-factor-eight-equivalent reranking budget, token top-k recovers 93.93 percent of the full-token score, while greedy marginal-gain selection recovers 98.39 percent. Held-out and leave-one-dataset-out evaluations show positive greedy improvements over token top-k on every dataset. Latency analysis identifies two useful operating points: token top-k for interactive retrieval and greedy selection as a quality upper envelope. The results show that late-interaction visual retrieval should be compressed by query-aware allocation, not only by query-agnostic pooling.**

***Index Terms*—visual document retrieval, late interaction, token selection, submodular optimization, efficient retrieval**

I. Introduction

Document retrieval is increasingly applied to corpora whose meaning is not captured by text alone: forms, scientific papers, financial reports, charts, invoices, slide decks, government documents, and multilingual technical pages. These documents are visual objects. Their semantics are distributed across words, tables, figures, typography, layout, and spatial grouping. Text-only pipelines based on optical character recognition, layout parsing, captioning, and chunking, including dedicated document-AI models such as LayoutLM [1] and OCR-free transformers such as Donut [2], can be useful, but they often discard or distort evidence that is easy for a human reader to see on the page.

Recent visual document retrievers address this limitation by embedding document pages directly. ColPali combines visual page embeddings with a ColBERT-style MaxSim operator and outperforms conventional OCR-based pipelines on the ViDoRe benchmark [3], [4]. Complementary approaches include single-vector Document Screenshot Embedding [5] and the visual retrieval-augmented generation pipeline VisRAG [6], both of which encode page screenshots without parsing. ModernVBERT and ColModernVBERT further show that compact retrieval-oriented vision-language encoders, built on contrastive image-text pretraining with sigmoid losses similar to SigLIP [7], can preserve much of this quality at lower model cost [8]. These models make visual retrieval simpler and more accurate, but they also expose a new systems bottleneck: each page is represented by hundreds of token vectors rather than one vector.

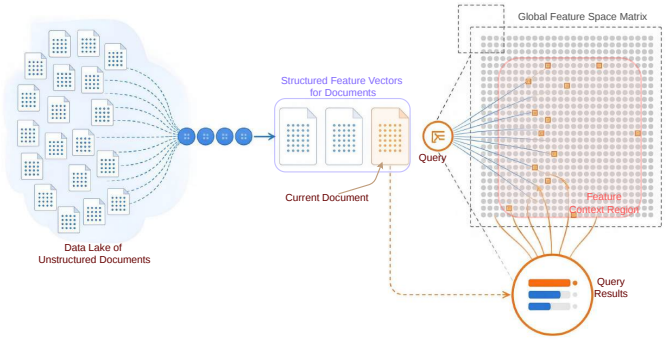

Fig. 1. Query-aware token budgeting selectively retains relevant document tokens for efficient late-interaction retrieval.

The cost is structural. For query token embeddings $Q = \{q_i\}_{i=1}^{m}$ and document token embeddings $D = \{d_j\}_{j=1}^{n}$, a late-interaction retriever scores

$$\mathrm{MaxSim}(Q, D) = \sum_{i=1}^{m} \max_{1 \le j \le n} q_i^{\top} d_j. \tag{1}$$

The maximum over document tokens is precisely what makes the model effective on localized evidence such as table cells, captions, and figure labels. It is also what makes storage and scoring expensive. In our ColModernVBERT setting, a 768-pixel page has roughly 350 document tokens. A million-page index therefore requires hundreds of millions of token vectors before any query-time computation is considered.

A common response is static token pooling: cluster or aggregate the page tokens once at indexing time and store only the representatives [9]. Related approaches include token pruning that discards uninformative embeddings [10], [11] and product-quantization-based engines that compress token residuals after centroid clustering [12]–[14]. Static pooling is attractive because it reduces the full index. However, it must

serve all future queries using a single compressed representation. This is misaligned with MaxSim. The token that matters for a query is not necessarily globally salient on the page; it is the token that best covers a particular query term. If pooling merges or removes that token, the query-time maximum can change even when the compressed page representation remains geometrically plausible.

This paper studies a different allocation of computation. We decouple corpus-scale candidate generation from query-specific evidence preservation. Stage one uses a strongly compressed index to retrieve a small candidate set, in the spirit of two-stage neural ranking pipelines that pair an efficient first retriever with a more expensive verifier [15]. Stage two revisits the original token set only for those candidate pages and selects a fixed budget of tokens for final MaxSim scoring (Fig. 2). The problem becomes: given a query, a candidate page, and a token budget, which document tokens should be retained?

This view turns compression into a data mining problem over multi-vector representations: mining a compact subset of document tokens that collectively covers the query under a strict budget. It also creates a natural link to submodular coverage. Adding a token that covers an already-covered query aspect has smaller marginal value than adding one that covers a missing aspect. This diminishing-returns structure suggests that budgeted token selection should reward complementarity, not only independent relevance, mirroring the use of submodular coverage and maximal marginal relevance in classical information retrieval and summarization [16]–[18].

We make the following contributions.

- We formulate query-aware token budgeting for late-interaction visual document retrieval as a stage-two MaxSim coverage problem under a per-page cardinality budget.
- We analyze the clipped token-selection objective as a monotone submodular function and use this structure to define a greedy marginal-gain selector.
- We compare random, uniform, cluster-guided Budget-Constrained Reranking (BCR), token top-k, and greedy selection under matched budgets on ten ViDoRe tasks using ColModernVBERT.
- We report quality, recovery, held-out generalization, leave-one-dataset-out transfer, and latency measurements, showing a clear frontier between token top-k and greedy selection.

The main empirical result is that static compression alone is not enough. Direct pooling from roughly 350 tokens per page to approximately 11 tokens per page drops macro nDCG@5 from 0.6309 to 0.4738. Under the same compressed first-stage regime, a pool-factor-eight-equivalent second-stage budget recovers most of the lost quality: token top-k reaches 0.5926 and greedy marginal-gain selection reaches 0.6208, or 98.39 percent of the uncompressed baseline.

II. Related Work

A. Text and dense retrieval

Classical information retrieval relies on sparse lexical signals such as TF-IDF and BM25 [19], [20]. Neural bi-encoders such as Sentence-BERT and Dense Passage Retrieval encode queries and passages into single dense vectors, enabling efficient approximate nearest-neighbor search with libraries such as FAISS [21]–[23]. Learned sparse retrievers such as SPLADE recover lexical controllability while remaining trainable end-to-end [24]. To recover finer-grained relevance signals at a manageable cost, BERT-style cross-encoders are commonly used to rerank a small candidate set produced by a faster first stage [15]. These models are scalable but can blur fine-grained evidence inside a document. Retrieval-augmented generation further raises the stakes: generation quality depends on whether the retriever surfaces the right evidence in the first place [25].

B. Late interaction

Late-interaction retrieval, exemplified by ColBERT, stores a set of contextual token embeddings for each document and scores a query by summing each query token's best document-token match [4]. ColBERTv2 introduced residual compression and a denoised supervision strategy [12], while PLAID further accelerated retrieval with centroid interaction and centroid pruning [13]. EMVB augments PLAID with optimized bit-vector prefiltering and SIMD-accelerated centroid interaction [14]. XTR rethinks the role of token retrieval by training the encoder to surface important document tokens early, sidestepping the cost of materializing all token vectors at scoring time [26]. ColBERTer reduces the stored vector count by aggregating subword tokens into whole-word representations and removing those that are not essential to scoring [27]. Our work is complementary: rather than reducing dimensionality, learning sparser token retrieval, or optimizing inverted-list probing, we study how to allocate a fixed number of *original* document tokens during stage-two reranking.

C. Visual document retrieval

Vision-language models such as CLIP and SigLIP established shared image-text embedding spaces [7], [28], but document retrieval requires finer visual grounding than natural-image retrieval. Earlier document-AI work approached this via OCR-then-language-model pipelines: LayoutLM jointly pretrains text and layout for document image understanding [1], and Donut removes OCR entirely with an end-to-end encoder-decoder [2]. ColPali introduced direct visual page retrieval with late interaction and the ViDoRe benchmark, showing that page images can be retrieved without lossy OCR-first pipelines [3]. Document Screenshot Embedding pursues a similar paradigm with a single-vector encoder [5], while VisRAG extends visual retrieval into retrieval-augmented generation by directly embedding pages as images [6]. ModernVBERT and ColModernVBERT reduce the model-size burden while preserving late-interaction document retrieval performance [8]. These models shift the deployment bottleneck from encoder size to token storage and MaxSim scoring, motivating our focus on token budgets.

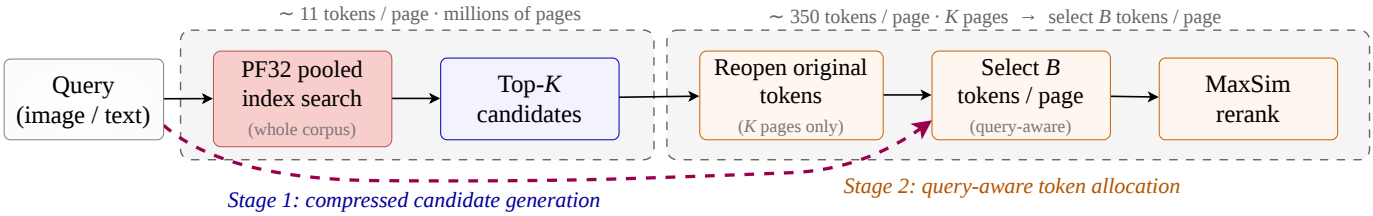

Fig. 2. Two-stage retrieval pipeline. Stage 1 (blue) searches a PF32-pooled index across the full corpus to retrieve K candidate pages, storing $\approx$11 tokens per page and reducing hot-path storage and interaction cost by 32×. Stage 2 (orange) reopens the original $\approx$350-token representations for those K pages and applies query-aware token budgeting to select a budget of B tokens per page before final MaxSim reranking. The key asymmetry is that Stage 1 is query-agnostic and corpus-wide, while Stage 2 is query-specific and restricted to the shortlist.

D. Token reduction and coverage selection

Token reduction methods include hard pruning, clustering, pooling, and merging. In efficient transformer inference, EViT prunes uninformative vision tokens [29], DynamicViT learns a token sparsification policy end-to-end [30], and ToMe merges similar tokens to retain object-level structure without retraining [31]. In multi-vector retrieval, token pruning preserves the highest-value embeddings per document [10], [11], while clustering-based pooling merges similar tokens within a document and is the source of the pool-factor (PF) terminology we adopt: PF= p reduces vector count by a factor of p with minimal quality loss for small p [9]. PLAID and EMVB approach the same problem from a quantization angle, storing centroid identifiers plus low-bit residuals rather than full token vectors [13], [14]. The common limitation of these query-agnostic methods is that compression cannot know which token a later query will need. Our work instead uses pooling for coarse candidate generation and shifts fine-grained allocation to query time.

Submodular optimization provides a principled language for coverage under budgets. For monotone submodular maximization with a cardinality constraint, greedy selection obtains the classical $(1 - 1/e)$ approximation guarantee [32], with broader treatments and applications surveyed in [18]. Lin and Bilmes used submodular coverage and diversity terms to obtain principled extractive summaries [17]; the closely related Maximal Marginal Relevance criterion of Carbonell and Goldstein combines query relevance and information novelty for diversity-aware reranking [16]. Recent dense retrieval work such as DISCO casts multi-vector retrieval as collective query coverage and uses submodular structure to guide efficient subset selection at the corpus level [33]. We adapt the same coverage idea within a single visual document page: the selected set consists of document tokens, not corpus items.

III. Query-Aware Token Budgeting

A. Problem setting

For each candidate page d, let $D = \{d_1, \ldots, d_n\} \subset \mathbb{R}^r$ be the original document-token set and let $Q = \{q_1, \ldots, q_m\} \subset \mathbb{R}^r$ be the query-token set. Static pooling [9] stores a query-agnostic representation $\widetilde{D}_p$ with about n/p tokens, where p is the pool factor. In our two-stage pipeline, stage one searches a PF32 index to obtain a candidate set $C_K(Q)$. Stage two reopens each $d \in C_K(Q)$ and selects a subset $S \subseteq D$ with $|S| \le B$, where B corresponds to a target pool-factor-equivalent budget such as PF8, as illustrated in Fig. 2.

Storage architecture. This design maintains two storage tiers. The PF32 pooled index (≈ 11 tokens per page) resides on fast storage and handles first-stage retrieval across the full corpus; its 32× footprint reduction lowers both the stored vector count and first-stage MaxSim interaction FLOPs by the same factor. The full token set (≈ 350 tokens per page) is retained on slower or cold storage and is loaded only for the K shortlisted candidates per query. The net result is a 32× reduction in hot-path index size, at the cost of keeping original tokens accessible for stage-two reranking. This contrasts with static pooling, which eliminates the full-token store entirely but cannot recover discarded evidence at query time. The ideal stage-two subset maximizes the retained MaxSim score:

$$S^*(Q, D; B) = \operatorname*{arg\,max}_{S \subseteq D,\, |S| \le B} \sum_{i=1}^{m} \max_{d \in S} q_i^\top d. \tag{2}$$

This objective is query-aware because the selected subset changes with Q. It is also coverage-oriented: a good subset must support all informative query tokens, not merely contain individually high-scoring page tokens.

B. Selection policies

We compare five policies that differ in query awareness and redundancy awareness. *Random reopening* spends the budget without using the query or page structure. It is a diagnostic lower bound. *Uniform reopening* spreads the budget broadly across the compressed representation or token groups. It is query-agnostic but coverage-preserving, and therefore a stronger baseline than random selection. *Budget-Constrained Reranking* (BCR) combines a uniform coverage backbone with query-guided cluster reopening. Let $\rho \in [0, 1]$ be the coverage ratio. Approximately ρB tokens are allocated uniformly, and the remaining budget is assigned to clusters whose members show higher query affinity. BCR tests whether coarse cluster-level query guidance is sufficient. *Token top-k* scores each original document token independently by its best query-token similarity,

$$\sigma(d \mid Q) = \max_{1 \le i \le m} q_i^\top d, \tag{3}$$

then keeps the B highest-scoring tokens. This is fast and strongly query-aware, but it does not penalize redundancy:

Algorithm 1 Greedy Marginal-Gain Token Selection

Require: Query tokens $Q = \{q_i\}_{i=1}^{m}$; document tokens $D = \{d_j\}_{j=1}^{n}$; budget B
Ensure: $S \subseteq D$ with $|S| \le B$

$\phi_i(d) \leftarrow \max\{0,\, q_i^\top d\}$ for all i, d ▷ clipped similarity
$S \leftarrow \emptyset$; $c_i \leftarrow 0$ for all i ▷ per-query-token coverage
for $t = 1, \ldots, B$ **do**
  for each $d \in D \setminus S$ **do**
    $\Delta(d) \leftarrow \sum_{i=1}^{m} \max\{0,\, \phi_i(d) - c_i\}$
  end for
  $d^* \leftarrow \arg\max_{d \in D \setminus S} \Delta(d)$
  $S \leftarrow S \cup \{d^*\}$; $c_i \leftarrow \max\{c_i,\, \phi_i(d^*)\}$ for all i
end forreturn S

Algorithm 2 Token Top-k Selection

Require: Query tokens $Q = \{q_i\}_{i=1}^{m}$; document tokens $D = \{d_j\}_{j=1}^{n}$; budget B
Ensure: $S \subseteq D$ with $|S| = B$

for each $d \in D$ **do**
  $\sigma(d) \leftarrow \max_{1 \le i \le m} q_i^\top d$
end forreturn top-B elements of D ranked by $\sigma(\cdot)$

several selected tokens may cover the same query aspect, much as classical relevance-only rerankers select redundant documents before MMR-style diversification [16].

Greedy marginal-gain selection directly optimizes coverage. Starting with $S_0 = \emptyset$, at step t it chooses

$$d_{t+1}^* = \arg\max_{d \in D \setminus S_t} \left[F_Q(S_t \cup \{d\}) - F_Q(S_t) \right], \tag{4}$$

where $F_Q(S)$ is the MaxSim-style coverage utility defined below. This method is redundancy-aware because a token receives credit only for the query-token coverage it adds beyond the current set.

Algorithms 1 and 2 give complete pseudocode for the two primary policies. Table I summarizes per-page computational costs; the observed stage-two latency ratio of $\approx 34\times$ for Greedy over Token top-k (Table V) is consistent with the $O(B)$ multiplicative factor at $B \approx 45$.

C. Submodular structure

The raw inner product in (1) may be negative. For the selection objective, we use the standard nonnegative coverage version obtained by including a dummy zero token, equivalently $\phi_i(d) = \max\{0, q_i^\top d\}$. Define

$$F_Q(S) = \sum_{i=1}^{m} \max_{d \in S} \phi_i(d), \qquad F_Q(\emptyset) = 0. \tag{5}$$

The final reranking score is computed with the selected tokens using the ordinary MaxSim operator.

Proposition 1. *For a fixed query Q, the set function $F_Q(S)$ in (5) is monotone submodular over document-token subsets. Therefore, greedy selection under a cardinality budget B achieves the classical $(1 - 1/e)$ approximation guarantee for maximizing F_Q.*

Proof. For a single query token q_i, define $f_i(S) = \max_{d \in S} \phi_i(d)$ with $f_i(\emptyset) = 0$. If $A \subseteq B$, then $f_i(A) \le f_i(B)$, so f_i is monotone. For any token $x \notin B$, the marginal gain is

$$f_i(A \cup \{x\}) - f_i(A) = \max\{0, \phi_i(x) - f_i(A)\}. \tag{6}$$

Since $f_i(A) \le f_i(B)$, we have $\phi_i(x) - f_i(A) \ge \phi_i(x) - f_i(B)$, and therefore $\max\{0, \phi_i(x) - f_i(A)\} \ge \max\{0, \phi_i(x) - f_i(B)\}$, i.e., $f_i(A \cup \{x\}) - f_i(A) \ge f_i(B \cup \{x\}) - f_i(B)$. This is the submodularity (diminishing-returns) condition, so f_i is submodular. A nonnegative sum of monotone submodular functions is monotone submodular, so $F_Q = \sum_i f_i$ is monotone submodular. The greedy guarantee follows from Nemhauser et al. [32]; see also the survey of submodular function maximization in [18]. □

The proposition clarifies the difference between token top-k and greedy selection. Token top-k estimates token salience independently. Greedy estimates marginal coverage, which is the relevant quantity under MaxSim when tokens can be redundant.

Remark (selection vs. scoring objective). Algorithm 1 maximizes the clipped objective F_Q defined in (5) to ensure submodularity holds. Final reranking scores in Tables V–IX are computed with the standard MaxSim operator (1), which permits negative inner products. The two objectives agree on which tokens are selected: a document token d with $q_i^\top d < 0$ for all i contributes zero marginal gain to F_Q and is never chosen by the greedy procedure. Negative-scoring tokens are therefore filtered implicitly during selection, and the clipped objective is a sound surrogate for the unclipped scoring criterion.

TABLE I
PER-PAGE TIME COMPLEXITY AT STAGE TWO. n: DOCUMENT TOKENS; m: QUERY TOKENS; B: BUDGET; k: CLUSTERS ($k \approx n/32$ UNDER PF32 INDEXING).

Method	Time	Space
Random	$O(B)$	$O(1)$
Uniform	$O(n)$	$O(1)$
Token top-k	$O(nm)$	$O(n)$
BCR	$O(km + n)$	$O(k)$
Greedy (naive)	$O(Bnm)$	$O(n{+}m)$
Lazy greedy	$O\big((n{+}B)m \log n\big)$ exp.	$O(n)$

TABLE II
SELECTION POLICIES COMPARED IN THE RERANKING STAGE.

Policy	Query-aware	Redundancy-aware	Unit
Random	No	No	token/cluster
Uniform	No	Implicit	token/cluster
BCR	Yes	Coarse	cluster
Token top-k	Yes	No	token
Greedy	Yes	Yes	token

TABLE III
VIDORE TASKS USED IN THE EVALUATION [3]. DOCVQA [34], INFOVQA [35], AND TAT-DQA [36] HAVE DEDICATED DATASET PAPERS.

Dataset	Queries	Docs	Characteristic
DocVQA	500	500	scanned industrial docs
InfoVQA	500	500	web infographics
TAT-DQA	1600	1600	financial tables
ArXivQA	500	500	scientific figures
TabFQuAD	210	210	French web tables
Energy	100	1000	energy reports
Government	100	1000	administrative pages
Healthcare	100	1000	medical pages
AI	100	1000	AI scientific pages
Shift Project	100	1000	French environmental reports

IV. EXPERIMENTAL SETUP

A. Benchmark and model

We evaluate on ten ViDoRe tasks [3]: DocVQA [34], InfoVQA [35], TAT-DQA [36], ArXivQA, TabFQuAD, Energy, Government, Healthcare, Artificial Intelligence, and Shift Project. The benchmark spans scanned industrial documents, infographics, financial tables, scientific figures, web tables, government pages, medical pages, energy reports, AI-related scientific pages, and French environmental reports. Query counts range from 100 for the practical tasks to 1600 for TAT-DQA.

All experiments use ColModernVBERT [8] as the base visual retriever. It preserves the multi-vector late-interaction structure of ColPali while using a compact 250M-parameter encoder. In the model comparison reported with ModernVBERT, ColModernVBERT obtains a ViDoRe average of 68.6 with CPU query-encoding latency of 0.032 seconds, compared with 69.2 and 0.222 seconds for ColPali [8]. This makes token storage and interaction cost the main efficiency issue.

B. Budgets and metrics

The uncompressed PF1 representation has about 350 page tokens. Static compression following [9] stores pooled representations at PF4, PF8, PF16, and PF32, corresponding roughly to 88, 45, 22, and 11 tokens per page. The main two-stage experiments use PF32 for candidate generation and a PF8-equivalent reranking budget unless otherwise stated. The full ten-dataset design-space comparison uses $K = 50$ candidates; held-out and latency sensitivity analyses additionally consider $K = 20$ and $K = 100$.

The main quality metric is macro-averaged nDCG@5 over datasets. We also report recovery relative to the uncompressed PF1 score:

$$\text{Recovery}(M) = 100 \cdot \frac{nDCG@5(M)}{nDCG@5(\text{PF1})}. \tag{7}$$

Latency is reported as median stage-two time and total median query time on the measured latency subset.

TABLE IV
DIRECT STATIC POOLING ON TEN VIDORE TASKS.

Pool factor	Tokens/page	Macro nDCG@5	Drop vs. PF1
PF1	350	0.6309	0.00%
PF4	88	0.6003	4.85%
PF8	45	0.5696	9.72%
PF16	22	0.5193	17.68%
PF32	11	0.4738	24.89%

TABLE V
DESIGN-SPACE COMPARISON AT PF8-EQUIVALENT BUDGET AND $K = 50$. RECOVERY IS RELATIVE TO PF1.

Method	Macro nDCG@5	Recovery	Stage-2 ms
Random	0.2332	36.97%	12.428
Uniform	0.4790	75.93%	10.550
BCR ($\rho = 0.75$)	0.5333	84.53%	30.736
Token top-k	0.5926	93.93%	15.832
Greedy	0.6208	98.39%	538.480

V. RESULTS

A. Static pooling has a hard ceiling

Table IV gives the direct-compression baseline. Performance decreases monotonically with pool factor. At PF8, direct pooling retains 90.28 percent of PF1; at PF32, it retains only 75.11 percent. This establishes the core failure mode of query-agnostic compression: an index that is highly efficient for candidate generation is too lossy for final ranking. This pattern is consistent with the original findings on clustering-based token pooling [9], although our ratios are computed against a visual late-interaction backbone rather than a text one.

The left panel of Fig. 3 visualizes the same trend. The sharp PF32 loss motivates a two-stage design: use PF32 to cheaply find candidates, but do not trust its compressed representation as the final evidence for ranking, as is also standard practice in neural reranking pipelines [15].

B. Query-aware token budgeting recovers quality

Table V compares selection policies at a PF8-equivalent stage-two budget. The ordering is consistent with the coverage interpretation: random $<$ uniform $<$ BCR $<$ token top-k $<$ greedy. Random selection collapses because spending a budget without structure rarely preserves the relevant evidence. Uniform reopening is much stronger, showing that broad coverage matters even without query information. BCR improves over uniform by adding coarse query guidance. Token top-k improves further by using token-level query affinity. Greedy gives the best score by accounting for redundancy through marginal gain.

The improvement from token top-k to greedy is especially informative. Both methods use query information at token granularity. The difference is that greedy discounts tokens whose query aspects have already been covered. The gain from 0.5926 to 0.6208 therefore reflects redundancy control rather than access to more query information, paralleling the

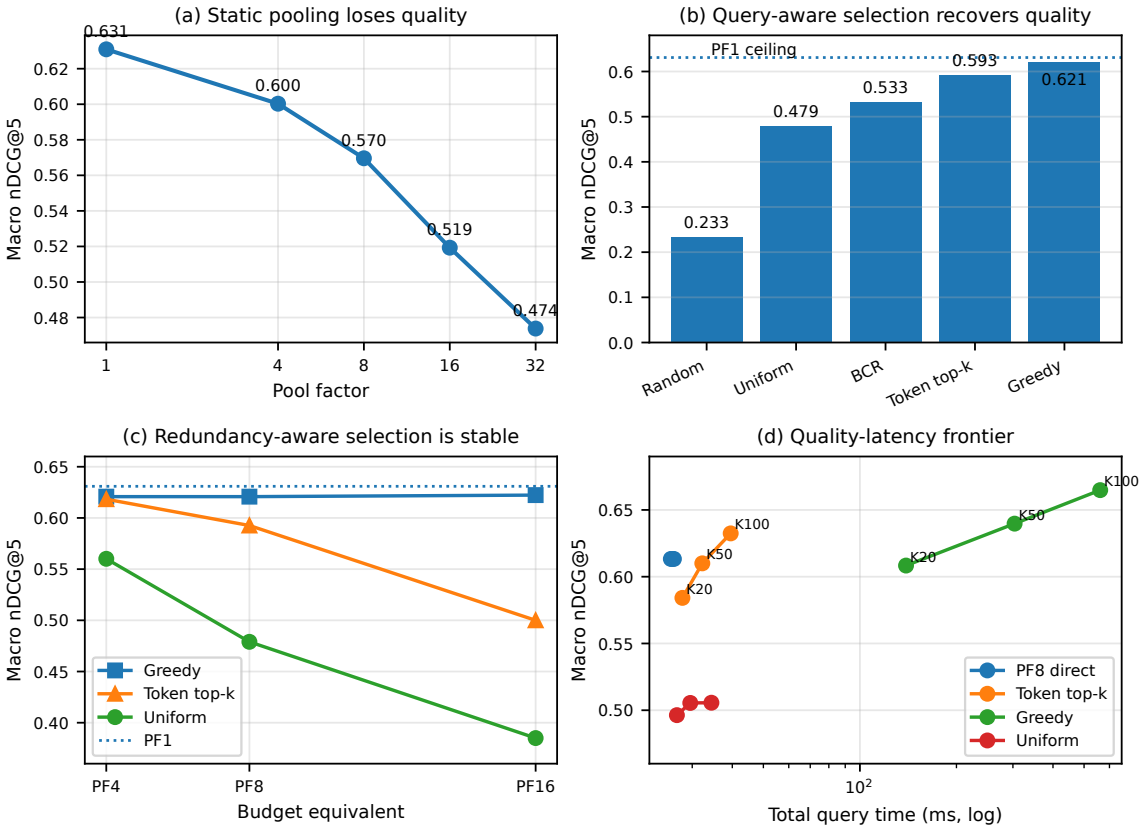

Fig. 3. Main empirical trends. Static pooling loses quality as compression increases. Query-aware selection recovers most of the PF1 score under a PF8-equivalent budget. Greedy is stable across budgets because it selects by marginal coverage. Token top-k is the strongest low-latency operating point, while greedy defines the quality upper envelope.

TABLE VI
SELECTION QUALITY ACROSS RERANKING BUDGETS AT $K = 50$.

Budget	PF1	Greedy	Top-k	Uniform
PF4	0.6309	0.6208	0.6183	0.5601
PF8	0.6309	0.6208	0.5926	0.4790
PF16	0.6309	0.6223	0.5000	0.3850

gains diversity-aware criteria such as MMR [16] obtain over relevance-only rankers and the role of submodular diversity in extractive summarization [17].

C. Greedy is robust across budgets

Table VI evaluates PF4-, PF8-, and PF16-equivalent reranking budgets. Greedy remains essentially flat across this range, staying near 0.621. Token top-k is competitive at PF4 but degrades at PF8 and PF16, while uniform reopening degrades more sharply. This suggests that independent token affinity works when the budget is generous enough to tolerate redundancy, but marginal-gain selection becomes increasingly valuable as the budget tightens.

D. Cluster-level guidance helps but is limited

BCR interpolates between uniform coverage and query-guided cluster reopening. Fig. 4 shows that pure query-guided cluster reopening is brittle, while near-pure coverage underuses the query. The best setting is $\rho = 0.75$, where most of the budget protects broad coverage and the remaining budget adapts to the query. This confirms that cluster-level query information is useful, but still less flexible than token-level selection.

E. Held-out and transfer results

To test whether the greedy advantage is a product of tuning to the full benchmark, we tune on three validation datasets and evaluate on seven held-out datasets. Greedy reaches 0.6460 macro nDCG@5, while token top-k reaches 0.6195. Relative to the held-out PF1 ceiling of 0.649, this is 99.54 percent recovery for greedy and 95.46 percent for token top-k. Greedy wins on all seven held-out datasets. A Wilcoxon signed-rank test gives $p = 0.0156$, a paired t test gives $p = 0.0358$, and a bootstrap 95 percent confidence interval for the mean delta is $[0.0096, 0.0453]$.

Fig. 5 shows the per-dataset deltas. The largest held-out gains occur on ArXivQA and Shift Project, both settings where evidence can be distributed across visually distinct regions. Leave-one-dataset-out transfer over all ten datasets gives the

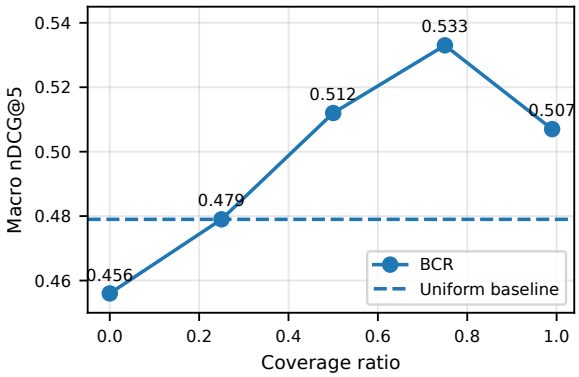

Fig. 4. BCR coverage-ratio ablation at PF8-equivalent budget and $K = 50$. The best value occurs at $\rho = 0.75$, indicating that coverage and query guidance are complementary.

TABLE VII
HELD-OUT SEVEN-DATASET COMPARISON AFTER TUNING ON A DISJOINT VALIDATION SPLIT.

Method	Macro nDCG@5	Recovery	Wins
Greedy	0.6460	99.54%	7/7
Token top-k	0.6195	95.46%	0/7

same qualitative result: greedy wins on every held-out condition, with mean delta 0.0282, minimum delta 0.0026, and maximum delta 0.0603.

TABLE VIII
PER-DATASET GREEDY VERSUS TOKEN TOP-k DELTAS. THE HELD-OUT COLUMNS REPORT THE LOCKED SEVEN-DATASET PROTOCOL; THE TRANSFER COLUMNS REPORT LEAVE-ONE-DATASET-OUT EVALUATION. DASHES INDICATE DATASETS USED IN THE VALIDATION SPLIT RATHER THAN THE SEVEN-DATASET HELD-OUT SPLIT.

Dataset	Held-out protocol			Leave-one-out transfer		
	Greedy	Top-k	Δ	Greedy	Top-k	Δ
ArXivQA [3]	0.7044	0.6388	0.0656	0.6946	0.6343	0.0603
Healthcare [3]	0.7939	0.7628	0.0311	0.8021	0.7533	0.0488
Shift Project [3]	0.5574	0.4991	0.0584	0.5372	0.4891	0.0482
InfoVQA [35]	–	–	–	0.6761	0.6429	0.0331
TabFQuAD [3]	0.4202	0.4066	0.0136	0.4149	0.3852	0.0297
AI [3]	–	–	–	0.8427	0.8136	0.0291
Government [3]	0.7790	0.7720	0.0069	0.7993	0.7824	0.0169
Energy [3]	0.8133	0.8076	0.0057	0.8016	0.7946	0.0070
TAT-DQA [36]	–	–	–	0.1967	0.1906	0.0061
DocVQA [34]	0.4540	0.4500	0.0040	0.4424	0.4398	0.0026

F. Latency-quality trade-off

Table IX reports the measured latency profile on the three-dataset latency subset at $K = 100$. Greedy gives the best quality but is much slower because it recomputes marginal gains sequentially. Token top-k is much faster: it improves over PF8-direct in quality on this subset while keeping total median latency below 40 ms. This identifies two useful operating points. Token top-k is the practical interactive default, while greedy is useful as a latency-tolerant quality ceiling or as a teacher for future approximate selectors.

TABLE IX
LATENCY-QUALITY TRADE-OFF ON THE THREE-DATASET LATENCY SUBSET AT $K = 100$. TOTAL TIME INCLUDES BOTH STAGES.

Method	Macro nDCG@5	Stage-2 ms	Total ms
PF8-direct	0.6133	–	25.678
PF32 + Uniform	0.5056	10.550	34.490
PF32 + Random	0.2270	12.428	36.366
PF32 + Token top-k	0.6324	15.832	39.776
PF32 + BCR	0.5091	30.736	54.671
PF32 + Greedy	0.6648	538.480	562.443

The latency sweep in Fig. 3 and Table X further shows that token top-k remains on the low-latency frontier as K varies, while greedy moves to much higher latency but also higher quality. The result does not suggest replacing all interactive retrieval with greedy. Instead, it shows that query-aware token budgeting has a useful family of operating points, complementing existing engine-level optimizations such as PLAID and EMVB [13], [14].

VI. DISCUSSION

The experiments support three lessons.

First, static pooling is useful for candidate generation but insufficient as the final representation under aggressive budgets. PF32 indexing is attractive because it stores roughly 11 tokens per page, but its standalone score is 0.4738. Reopening shortlisted pages and spending a modest query-aware budget recovers much of the lost quality. This complements existing static-compression work [9]–[11], [27], which improves the first stage but does not address the final ranking ceiling.

Second, the key resource is not token count alone but query coverage. Uniform reopening demonstrates the value of broad coverage; BCR demonstrates the value of query guidance; token top-k demonstrates the value of token-level affinity; greedy demonstrates the additional value of redundancy control. The monotonic ordering of these methods across the main experiment is exactly what the coverage view predicts, mirroring the long-standing insight in summarization and diversification that relevance plus novelty beats relevance alone [16], [17].

Third, the best method depends on latency requirements. Greedy is not the deployment answer for every system. Its value is that it defines a principled quality envelope and shows how much quality is lost by independent token scoring. Token top-k is less theoretically complete but more practical, and it already recovers 93.93 percent of PF1 under the main ten-dataset protocol.

A. Theoretical bound versus empirical recovery

Proposition 1 guarantees $F_Q(S_{\text{greedy}}) \geq (1 - 1/e) \cdot F_Q(S^*)$ for the optimal size-B subset S^*, a worst-case bound of approximately 63%. The empirical *nDCG@5* recovery of 98.39% against the PF1 full-token baseline is substantially higher for two independent structural reasons.

First, the guarantee bounds $F_Q(S_{\text{greedy}})$ relative to the optimal *size-B subset* S^*, not relative to the full token set

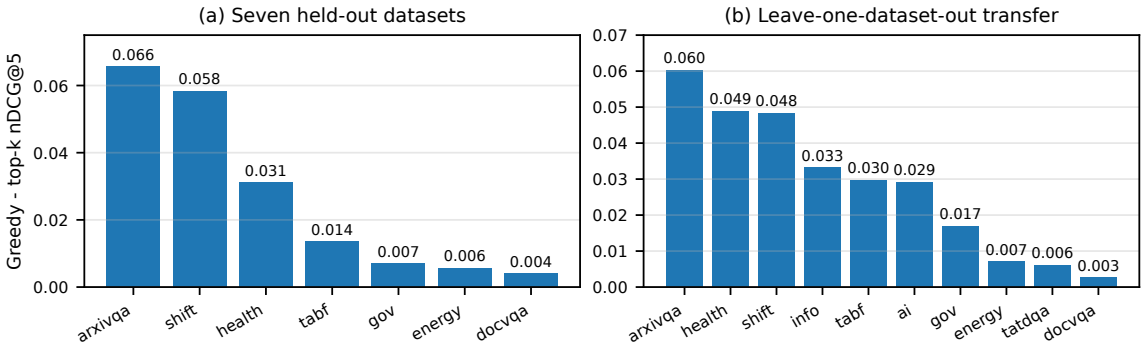

Fig. 5. Generalization of greedy over token top-k. Left: all seven held-out deltas are positive after tuning on a disjoint validation split. Right: leave-one-dataset-out transfer produces positive deltas for all ten held-out conditions.

TABLE X
LATENCY SENSITIVITY ACROSS CANDIDATE-POOL SIZES. EACH CELL REPORTS MACRO NDCG@5/TOTAL MEDIAN MILLISECONDS.

Method	$K = 20$	$K = 50$	$K = 100$
PF8-direct	0.6133 / 26.301	0.6133 / 25.981	0.6133 / 25.732
PF32 + Uniform	0.4963 / 26.873	0.5055 / 29.584	0.5056 / 34.430
PF32 + Random	0.2646 / 27.277	0.2414 / 30.593	0.2270 / 36.301
PF32 + Token top-k	0.5842 / 27.958	0.6100 / 32.251	0.6324 / 39.542
PF32 + BCR	0.4878 / 31.101	0.4953 / 39.767	0.5091 / 54.384
PF32 + Greedy	0.6083 / 139.259	0.6397 / 303.635	0.6648 / 561.438

D. The 98.39% figure instead measures against PF1, whose advantage over any size-B subset is bounded by the marginal value of the $n - B$ excluded tokens. Once the greedy set covers the dominant query aspects, the remaining tokens contribute negligible additional value; the practical gap between $F_Q(S_{\text{greedy}})$ and $F_Q(D)$ is far smaller than the worst-case gap between $F_Q(S_{\text{greedy}})$ and $F_Q(S^*)$ permits.

Second, the adversarial instances that force the $(1 - 1/e)$ tightness require that every greedy selection maximally duplicates existing coverage. Visual page token sets contain many near-duplicate embeddings arising from background patches, whitespace, and repeated visual motifs; early greedy selections therefore cover distinct query aspects without duplication, and coverage loss per step is far below the worst-case allowance. Together, these factors explain why the $(1 - 1/e)$ guarantee functions as a rigorous theoretical anchor while empirical recovery substantially exceeds it.

B. Reproducibility

All benchmark tasks are public ViDoRe page-retrieval tasks, and the paper reports the retriever, pool factors, candidate-pool sizes, per-page budgets, BCR coverage setting, evaluation splits, latency units, and all aggregate result values used in the figures and tables. The anonymous source bundle contains the plotting data and LaTeX source used to generate the submitted PDF. Code for exact reranking execution and cached result files can be released after review without anonymization constraints.

C. Limitations

This study focuses on ColModernVBERT and ViDoRe-style page retrieval. The qualitative mechanism should apply to other multi-vector retrievers, including ColBERT-family text retrievers and engines such as PLAID, EMVB, and XTR [13], [14], [26], but cross-backbone validation remains important. The current greedy implementation is also computationally expensive; future work should investigate lazy greedy with marginal-gain upper bounds [37], candidate-pruned greedy, learned token selectors building on the dynamic-token-sparsification ideas of DynamicViT and ToMe [29]–[31], or distillation from greedy selections into fast token policies. Finally, our budget is fixed per candidate page. Adaptive budgets could spend fewer tokens on easy candidates and more on visually dense or ambiguous pages.

VII. CONCLUSION

Late-interaction visual document retrieval is accurate because it preserves fine-grained page evidence, but that evidence is expensive to store and score. This paper shows that the cost should not be addressed only by query-agnostic pooling. A compressed index can generate candidates, but final ranking benefits from query-aware token allocation on the original token set. Formulating this allocation as MaxSim coverage explains why redundancy-aware greedy selection recovers near-full quality, and why token top-k is a strong low-latency approximation. On ten ViDoRe tasks, greedy recovers 98.39 percent of the PF1 score under a PF8-equivalent reranking budget, while token top-k provides the practical latency-quality frontier. The broader conclusion is simple: efficient

late interaction is not only about storing fewer tokens; it is about selecting the right tokens for the query.

References

[1] Y. Xu, M. Li, L. Cui, S. Huang, F. Wei, and M. Zhou, "LayoutLM: Pre-training of text and layout for document image understanding," in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2020, pp. 1192–1200.

[2] G. Kim, T. Hong, M. Yim, J. Nam, J. Park, J. Yim, W. Hwang, S. Yun, D. Han, and S. Park, "OCR-free document understanding transformer," in *European Conference on Computer Vision (ECCV)*, 2022, pp. 498–517.

[3] M. Faysse, H. Sibille, T. Wu, B. Omrani, G. Viaud, C. Hudelot, and P. Colombo, "ColPali: Efficient document retrieval with vision language models," *arXiv preprint arXiv:2407.01449*, 2024.

[4] O. Khattab and M. Zaharia, "ColBERT: Efficient and effective passage search via contextualized late interaction over BERT," in *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2020, pp. 39–48.

[5] X. Ma, S.-C. Lin, M. Li, W. Chen, and J. Lin, "Unifying multimodal retrieval via document screenshot embedding," in *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2024.

[6] S. Yu, C. Tang, B. Xu, J. Cui, J. Ran, Y. Yan, Z. Liu, S. Wang, X. Han, Z. Liu, and M. Sun, "VisRAG: Vision-based retrieval-augmented generation on multi-modality documents," in *International Conference on Learning Representations (ICLR)*, 2025.

[7] X. Zhai, B. Mustafa, A. Kolesnikov, and L. Beyer, "Sigmoid loss for language image pre-training," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023, pp. 11 975–11 986.

[8] P. Teiletche, Q. Macé, M. Conti, A. Loison, G. Viaud, P. Colombo, and M. Faysse, "ModernVBERT: Towards smaller visual document retrievers," *arXiv preprint arXiv:2510.01149*, 2025.

[9] B. Clavié, A. Chaffin, and G. Adams, "Reducing the footprint of multi-vector retrieval with minimal performance impact via token pooling," *arXiv preprint arXiv:2409.14683*, 2024.

[10] C. Lassance, M. Maachou, J. Park, and S. Clinchant, "A study on token pruning for ColBERT," *arXiv preprint arXiv:2112.06540*, 2021.

[11] A. Acquavia, C. Macdonald, and N. Tonellotto, "Static pruning for multi-representation dense retrieval," in *Proceedings of the ACM Symposium on Document Engineering 2023*, 2023, pp. 1–10.

[12] K. Santhanam, O. Khattab, J. Saad-Falcon, C. Potts, and M. Zaharia, "ColBERTv2: Efficient and effective retrieval via lightweight late interaction," *arXiv preprint arXiv:2112.01488*, 2021.

[13] K. Santhanam, O. Khattab, C. Potts, and M. Zaharia, "PLAID: An efficient engine for late interaction retrieval," in *Proceedings of the 31st ACM International Conference on Information & Knowledge Management (CIKM)*, 2022, pp. 1747–1756.

[14] F. M. Nardini, C. Rulli, and R. Venturini, "Efficient multi-vector dense retrieval with bit vectors," in *Advances in Information Retrieval – 46th European Conference on Information Retrieval (ECIR)*, ser. LNCS, vol. 14609. Springer, 2024, pp. 3–17.

[15] R. Nogueira and K. Cho, "Passage re-ranking with BERT," *arXiv preprint arXiv:1901.04085*, 2019.

[16] J. G. Carbonell and J. Goldstein, "The use of MMR, diversity-based reranking for reordering documents and producing summaries," in *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, 1998, pp. 335–336.

[17] H. Lin and J. Bilmes, "A class of submodular functions for document summarization," in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL)*, 2011, pp. 510–520.

[18] A. Krause and D. Golovin, "Submodular function maximization," in *Tractability: Practical Approaches to Hard Problems*, L. Bordeaux, Y. Hamadi, and P. Kohli, Eds. Cambridge University Press, 2014, pp. 71–104.

[19] K. S. Jones, "A statistical interpretation of term specificity and its application in retrieval," *Journal of Documentation*, vol. 28, no. 1, pp. 11–21, 1972.

[20] S. Robertson, S. Walker, S. Jones, M. Hancock-Beaulieu, and M. Gatford, "Okapi at TREC-3," *NIST Special Publication SP 109*, 1995.

[21] N. Reimers and I. Gurevych, "Sentence-BERT: Sentence embeddings using siamese BERT-networks," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*, 2019, pp. 3982–3992.

[22] V. Karpukhin, B. Oguz, S. Min, P. Lewis, L. Wu, S. Edunov, D. Chen, and W.-t. Yih, "Dense passage retrieval for open-domain question answering," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, 2020, pp. 6769–6781.

[23] J. Johnson, M. Douze, and H. Jégou, "Billion-scale similarity search with GPUs," *IEEE Transactions on Big Data*, vol. 7, no. 3, pp. 535–547, 2019.

[24] T. Formal, B. Piwowarski, and S. Clinchant, "SPLADE: Sparse lexical and expansion model for first stage ranking," in *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2021, pp. 2288–2292.

[25] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. K"uttler, M. Lewis, W.-t. Yih, and T. Rockt"aschel, "Retrieval-augmented generation for knowledge-intensive NLP tasks," *Advances in Neural Information Processing Systems*, vol. 33, pp. 9459–9474, 2020.

[26] J. Lee, Z. Dai, S. M. K. Duddu, T. Lei, I. Naim, M.-W. Chang, and V. Y. Zhao, "Rethinking the role of token retrieval in multi-vector retrieval," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

[27] S. Hofstätter, O. Khattab, S. Althammer, M. Sertkan, and A. Hanbury, "Introducing neural bag of whole-words with ColBERTer: Contextualized late interactions using enhanced reduction," in *Proceedings of the 31st ACM International Conference on Information & Knowledge Management (CIKM)*, 2022, pp. 737–747.

[28] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, and J. Clark, "Learning transferable visual models from natural language supervision," in *Proceedings of the 38th International Conference on Machine Learning*, 2021, pp. 8748–8763.

[29] Y. Liang, C.-L. Ge, Z. Tong, Y. Song, J. Wang, and P. Xie, "EViT: Expediting vision transformers via token reorganizations," *International Conference on Learning Representations*, 2022.

[30] Y. Rao, W. Zhao, B. Liu, J. Lu, J. Zhou, and C.-J. Hsieh, "DynamicViT: Efficient vision transformers with dynamic token sparsification," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[31] D. Bolya, C.-Y. Fu, X. Dai, P. Zhang, C. Feichtenhofer, and J. Hoffman, "Token merging: Your ViT but faster," in *International Conference on Learning Representations (ICLR)*, 2023.

[32] G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher, "An analysis of approximations for maximizing submodular set functions," *Mathematical Programming*, vol. 14, no. 1, pp. 265–294, 1978.

[33] K. Nair, P. Chakraborty, A. Tambat, I. Roy, S. Chakrabarti, A. Dasgupta, and A. De, "A dense subset index for collective query coverage," *International Conference on Learning Representations*, 2026.

[34] M. Mathew, D. Karatzas, and C. V. Jawahar, "DocVQA: A dataset for VQA on document images," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2021, pp. 2200–2209.

[35] M. Mathew, V. Bagal, R. P. Tito, D. Karatzas, E. Valveny, and C. V. Jawahar, "InfographicVQA," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2022, pp. 1697–1706.

[36] F. Zhu, W. Lei, F. Feng, C. Wang, H. Zhang, and T.-S. Chua, "Towards complex document understanding by discrete reasoning," in *Proceedings of the 30th ACM International Conference on Multimedia (ACM MM)*, 2022, pp. 4857–4866.

[37] M. Minoux, "Accelerated greedy algorithms for maximizing submodular set functions," in *Optimization Techniques*, ser. Lecture Notes in Control and Information Sciences, J. Stoer, Ed., vol. 7. Springer Berlin Heidelberg, 1978, pp. 234–243.