EFFICIENT TEXT-IMAGE RETRIEVAL USING LARGE LANGUAGE MODELS

JIAHAO LIU

A THESIS SUBMITTED TO THE FACULTY OF GRADUATE STUDIES
IN PARTIAL FULFILMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
MASTER OF ARTS

GRADUATE PROGRAM IN INFORMATION SYSTEMS AND TECHNOLOGY
YORK UNIVERSITY
TORONTO, ONTARIO

DECEMBER 2025

Abstract

Efficient retrieval from large-scale image databases is a key challenge, particularly as applications increasingly rely on multimodal models such as CLIP. While CLIP offers strong joint image–text representations for semantic search, its globally pooled embeddings often struggle with fine-grained, multi-concept queries, leading to high false positives and reliance on costly verification models. To address this, we propose a hybrid framework that structures the embedding space through feature clustering and models candidate selection as a multi-armed bandit problem. Each cluster acts as an arm, with relevance scores from ground-truth systems as rewards. Using Thompson Sampling, this approach balances exploration and exploitation to quickly identify promising clusters, reducing unnecessary ground-truth queries. Experiments show that our method significantly improves precision and lowers computational costs in multi-keyword retrieval tasks, enabling scalable, fine-grained retrieval in resource-constrained settings. This structured, adaptive approach effectively enhances CLIP-based retrieval pipelines.

Acknowledgements

I wish to express my deepest gratitude to my supervisor, Professor Yu, for his consistent guidance, rigorous mentorship, and unwavering support throughout my research. His profound expertise and insightful advice greatly enhanced my understanding of the research domain and provided essential intellectual direction for this thesis. The opportunities he generously afforded me have significantly enriched my graduate education.

I am sincerely grateful to Professor Yueting for her invaluable guidance and contributions. His thoughtful advice and support were instrumental in helping me effectively integrate theoretical foundations with practical applications, and his mentorship has been a continual source of academic inspiration.

Finally, I would like to express my profound gratitude to my parents for their understanding, patience, and support throughout my academic journey. Their steadfast encouragement played a vital role in enabling me to persevere through this demanding yet deeply meaningful phase of my studies and has remained a constant source of strength.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Background and Motivation

The efficient management and retrieval of information from massive, real-world image databases remains a cornerstone challenge across disciplines, from large-scale e-commerce platforms to scientific data analysis. Modern solutions rely heavily on robust feature extraction by vision models and attempt to match such features with a query during query answering. While the Contrastive Language-Image Pre-training (CLIP) [16] model successfully maps images and text to a shared latent space, its limited ability to capture local and fine-grained information limits its use in many practical applications. Therefore, a critical bottleneck arises in scenarios that require querying an expensive, highly accurate "ground truth" model for validation after retrieving potential candidates from the CLIP global embedding. When a user query combines multiple distinct keywords (e.g., "a bike and a backpack"), the figure shows the

expected search results for the query. Correct result images are tagged with a check mark, and incorrect result images are tagged with a cross mark.

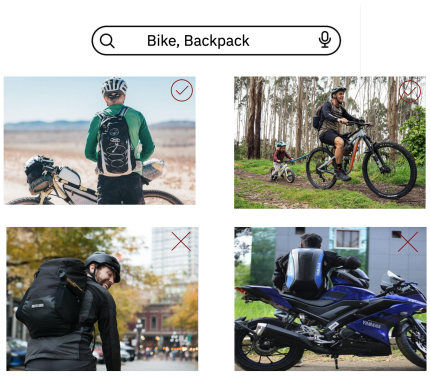

Figure 1.1: Image search for bike and backpack

However, the naive clip will rank these four images as follows.

Figure 1.2: Sorted order by naive clip embedding

One of the correct images is ranked after a wrong image. Because the backpack is not obviously visible in the correct image, the global embedding tends to average across the entire scene, and the embeddings are ranked after the wrong one, resulting in a high false-positive rate when precise relevance is required. This inefficiency severely limits the scalability of systems that rely on iterative feedback or resource-constrained validation.

1.2 Existing Approaches

The Contrastive Language-Image Pre-training (CLIP) model has emerged as a powerful existing tool, capable of generating rich, joint multi-modal embeddings that enable zero-shot image classification and highly semantic text-to-image matching. The dual-encoder architecture of CLIP makes it a good choice for image retrieval. Since it uses two independent encoders to map both modalities into a single shared embedding space, it enables pre-computation, allowing image embeddings to be computed and stored offline. Retrieval is then achieved by encoding the text query once and performing a lightning-fast vector similarity search against the pre-indexed database, making the system highly scalable and efficient for real-time applications. However, standard CLIP embeddings, typically derived via global average pooling, excel at capturing the overall theme but often fail in scenarios requiring fine-grained, compositional understanding.

Instead of a dual-encoder architecture, fusion-based models that employ cross-modal attention or LLMs are designed for tasks that require deep, token-level reasoning. These models require integrating image patches and text tokens into a single transformer for cross-attention-based interaction, ensuring fine-grained alignment. This dependence on query-specific fusion prevents feature pre-computation. For retrieval, the computationally expensive cross-attention operation must therefore be applied to every database image. This computational overhead makes fusion models prohibitively slow and inefficient for real-time, large-scale retrieval.

In the costly verification process, naive image selection methods, such as

simple random sampling or prioritizing candidates by embedding distance, often lead to overutilization of the expensive ground-truth model.

1.3 The Proposed Method

To address the limitations of the inefficient selection at a massive scale, our approach first structures the image database via clustering. This method ensures that each resulting cluster comprises semantically homogeneous images, ideally exhibiting a shared set of descriptive keywords.

We further frame the selection process as a multi-armed bandit (MAB) problem, where each cluster corresponds to an 'arm' and the 'reward' is defined by relevance feedback from the expensive ground-truth model. Thompson Sampling (TS) is an ideal heuristic for this MAB problem because it naturally balances the trade-off between exploration, which tries novel, under-sampled clusters, and exploitation, which focuses on clusters that have historically yielded high relevance scores. This probabilistic approach, which integrates uncertainty directly via Bayesian updating, ensures rapid convergence to the most valuable image clusters while preserving the ability to identify candidate images.

1.4 Contributions and Outline

In summary, we introduce a novel hybrid methodology for image selection based on a multi-keyword textual query. This approach combines feature clustering with Thompson Sampling to minimize reliance on an expensive

ground-truth system. To our knowledge, this is the first attempt to apply Thompson sampling to an image retrieval task. The remainder of this paper describes the problem definition and method details and presents experimental validation.

Chapter 2

Literature Review

2.1 Vision Language Models

Vision-Language Models(VLMs) represent a paradigm shift in artificial intelligence by combining the capabilities of computer vision and natural language processing. These models are designed to understand, reason, and generate content from joint visual and textual inputs, enabling complex tasks such as visual question answering (VQA), image captioning, and cross-modal retrieval. The modern era of VLM research is fundamentally anchored by the introduction of Contrastive Language-Image Pre-training (CLIP) [16] by OpenAI in 2021, which demonstrated unprecedented zero-shot generalization capabilities. The core of the CLIP model is a dual-encoder architecture comprising two independent neural networks: the image encoder and the text encoder. Image Encoder is typically a Vision Transformer (ViT) [6] or a ResNet [7] variant that processes the image input. The text encoder is a Transformer-based language

model [23] that processes the corresponding text caption.

Despite its strengths, the dual-encoder architecture of CLIP inherently limits the depth of interaction between modalities. Since image and text are encoded into separate, fixed-length embeddings before comparison, CLIP lacks the ability to attend to fine-grained information. This architecture is highly effective at aligning concepts but struggles with complex visual reasoning or visual question-answering tasks, which require detailed, token-level interaction—for instance, linking a specific object region in an image to a precise word in a question. Specifically, CLIP exhibits several critical weaknesses, including difficulty in accurately binding attributes to concepts within multi-object scenes [2, 9, 15], a tendency to misinterpret object layouts or conflate multiple entities in a single image [9, 15, 28], and poor performance when representing negation [20, 1]. Following CLIP, VLM research has diverged into two main streams: improving the efficiency of cross-modal alignment and integrating VLMs with Large Language Models (LLMs) for complex generative tasks.

To improve the efficiency of cross-modal alignment, FILIP (Fine-grained Interactive Language-Image Pre-training) [25] enhances the efficient dual-stream architecture of models such as CLIP. It extends beyond global feature matching by introducing a cross-modal late-interaction mechanism. This mechanism modifies the contrastive loss to achieve fine-grained semantic alignment by using token-wise maximum similarity between image patches and text tokens, while preserving CLIP's fast inference speed. Decoupled Contrastive Learning(DCL)[26] is a self-supervised objective that modifies the standard contrastive loss by removing the positive pair's term from the denominator. This

key change decouples positive and negative forces, eliminating the "positive-negative coupling" effect. As a result, DCL achieves high performance and training stability even with smaller batch sizes, unlike traditional methods that often require large batches. CoCa(Contrastive Captioner)[27] is an image-text foundation model designed to unify contrastive and generative pretraining paradigms into a single minimalist framework. This architecture enables efficient end-to-end pretraining of the model with a joint objective that combines a contrastive loss on unimodal embeddings with a captioning loss on multimodal outputs, thereby improving the representativeness of the embeddings. RegionCLIP[30] overcomes the poor performance of standard CLIP models on object detection tasks by introducing a pre-training framework that aligns image regions with synthetic text descriptions. It achieves this by using a fixed CLIP model as a “teacher” to generate “pseudo” labels for region-text pairs. This process trains the model to learn region-level representations, resulting in state-of-the-art zero-shot and open-vocabulary detection on benchmarks.

The advent of highly capable pre-trained LLMs, such as LLaMA[21] and ChatGPT, led to the next breakthrough: using these language models as the reasoning backbone for vision. Models in this category are often referred to as Visual Large Language Models (VLLMs) or Multimodal LLMs (MM-LLMs). BLIP-2(Bootstrapping Language-Image Pre-training with Latent Text)[10] introduces the Querying Transformer(Q-Former). This lightweight and parameter-efficient module connects a pre-trained image encoder to a frozen pre-trained LLM. The Q-Former explicitly uses cross-attention to query visual features from learned tokens, thereby addressing the fine-grained interaction

limitation of the dual-encoder architecture and enabling highly efficient integration without requiring expensive full-scale multimodal training. LLaVA (Large Language and Vision Assistant) [12] simplified the model architecture by adding a simple Multi-Layer Perceptron (MLP) projection layer that maps visual features from a frozen visual encoder directly into the input token space of a frozen LLM. LLaVA's key innovation was Visual Instruction Tuning, which trained the model on a small, high-quality dataset of multimodal instruction-following data, enabling it to become a powerful conversational assistant capable of complex visual reasoning and command execution.

2.2 CLIP-Based Image Retrieval

Image retrieval has progressed from basic keyword matching and Content-Based Image Retrieval (CBIR), which relied on low-level features like colour and texture, to advanced semantic retrieval. Historically, the core difficulty has been the semantic gap—the disconnect between the simple visual features that computers extract and the complex concepts that users actually seek. The introduction of CLIP marked a paradigm shift, effectively bridging this gap by learning a robust, general-purpose joint embedding space from massive datasets. Much follow-up research focuses on enhancing the CLIP model for image retrieval.

Some work focuses on fine-tuning and optimization explicitly designed for retrieval tasks. Schall et al. (2024)[19] address a key limitation of CLIP's inability to distinguish visually distinct images that share similar captions in

instance-level image retrieval. To boost visual precision without harming text retrieval, they propose two fine-tuning methods. Multi-Caption Image Pairing (MCIP) employs pseudo-captions and a novel Multi-Caption ArcMargin loss to enforce more substantial visual-text alignment. Experiments show that MCIP significantly improves instance-level and zero-shot image retrieval and classification, while maintaining strong text-to-image retrieval, enabling efficient, single-embedding multi-modal search.

Recent advancements have further refined these global image representations by incorporating local details from intermediate layers of the network. The DELG[3] model significantly aids image retrieval by resolving the traditional trade-off between retrieval accuracy and computational efficiency through the synergistic extraction of both global and local features within a single deep learning framework. It leverages compact global descriptors for efficiency, using them to provide an image signature that quickly narrows the search space in the initial retrieval stage. Additionally, DELG integrates local features to improve accuracy, facilitating the re-ranking or geometric verification stage, where fine-grained and patch-level details are matched to ensure that the retrieved image is genuinely relevant and to distinguish between highly similar or partially occluded images. The DOLG (Deep Orthogonal Local and Global feature fusion) [24] model is a single-stage image retrieval framework. It was designed to mitigate error accumulation and the computational burden of traditional two-stage retrieval systems by integrating local and global information into a single compact descriptor. The architecture features a local branch that uses convolutions and self-attention to extract discriminative local features. These

features are then fed into an orthogonal fusion module to isolate components orthogonal to the global image representation. By concatenating these complementary orthogonal local features with global features, DOLG generates a unified, optimized descriptor within a single end-to-end framework using the ArcFace [5] margin loss.

Chapter 3

Problem Definition

We consider an image repository X, where each element $x \in X$ is an image. To query the image, we treat the query Q as a set of keywords, where each keyword $q \in Q$ represents a single word or a phrase. For example, the keyword pair `{car, bike}` can be used to search for images that contain both a vehicle and a bike. Without loss of generality, we consider $\mathcal{V}$ as the set of possible keywords that could be used to describe the content from an image. We define our image retrieval problem on keywords as follows :

Definition 1 (Image Retrieval using Keywords)**.** *For a given set of keywords $Q \subseteq \mathcal{V}$, retrieve a set of images $R \subseteq X$, such that each image $r \in R$ is considered a match to the keywords Q.*

Such a problem has been widely studied and is known as image-to-text keyword retrieval [4]. In such work, they consider $\mathcal{V}$ is fixed or known a priori. In such cases, a classification or object detection model trained to recognize a predefined keyword can be used to produce the keywords associated with each

image. For example, 80 keywords representing object categories were used in the COCO dataset [11]). We refer to such a setting as a closed vocabulary, as the set $\mathcal{V}$ can be constructed before the query Q arrives. However, the closed-vocabulary setting limits users' search flexibility and requires passing the entire image dataset to models trained with new keywords, thereby continuously updating image tags to match them. This process is annoying and time-consuming.

In this problem, we focus on an open-vocabulary setting, where $|\mathcal{V}|$ is unknown or cannot be determined in advance. In such a setting, the set of possible keywords is unknown before queries arrive, making it impossible to rely on object detection or classification models that require a fixed set of prediction categories for each image.

Definition 2 (Oracle Model M)**.** *The oracle model (M) is defined as a function that determines whether an image x matches a query Q.*

The oracle model M is expected to return a "ground truth" answer about the relevance between an image x and a query Q. The "ground truth" doesn't imply that it is guaranteed to be absolutely correct; it is used only as a reference for the research. The choices of the oracle model M are arbitrary, ranging from advanced vision models to human annotators. In this research, we choose to use a Vision-Language model integrated with a Large Language Model as the oracle model M.

Formally, for a given image x and keyword query Q, we can verify whether x satisfies Q by leveraging large language models for vision. Let M be the model that produces a Boolean result for a given query Q on image x, then we

have $M(x, Q) \rightarrow \{0, 1\}$, where 1 indicates x is a match of query Q, and zero otherwise.

Definition 3 (Limit-k Image Retrieval)**.** *On an image repository X, given a keyword query Q, the objective of limit-k image retrieval is to retrieve a set of images $R \subseteq X$, such that for each $r \in R$, $M(r, Q) = 1$, and the size of set R is k, i.e., $|R| = k$.*

To perform an efficient limit-k image retrieval, we aim to construct an ordered candidates list X', such that $\forall x' \in X', x' \in X$, with model M, such that the model invocation to M is minimized.

Chapter 4

Method

In this section, we first present a baseline approach to solving the problem, along with optimization opportunities, in Section 4.1. We then discuss our solution to the problem, using Thompson Sampling from Section 4.2 to Section 4.2.2. Lastly, we introduce an image tiling technique to further optimize the performance in Section 4.3.

4.1 Baseline Approach

The baseline approach uses the CLIP model to precompute and store embeddings for all repository images as a matrix. Upon receiving a multi-keyword query, the ground-truth model selects images based on the distance between their embeddings and the query embedding.

4.1.1 CLIP Model

To enable direct semantic comparison and similarity measurement across image and text data modalities, a common embedding space must be constructed. Contrastive Language-Image Pre-training (CLIP) [16] is a widely recognized multi-modal model that achieves this by mapping both visual and textual inputs into a shared vector representation. CLIP is trained to learn semantic relationships between general concepts in images and text and to project these relationships into a shared vector space. The CLIP model will place image and text vectors in a shared vector space such that semantically similar items are close together. CLIP uses two dedicated encoders for images and texts. Let $E_I(\cdot)$ be the **Image Encoder** and $E_T(\cdot)$ be the **Text Encoder**. Both are parameterized neural networks that map their respective inputs to a D-dimensional embedding space.

For an input image x, the image embedding $\mathbf{v}$ is produced:

$$\mathbf{v} = E_I(x) \in \mathbb{R}^D$$

For an input text t, the text embedding $\mathbf{u}$ is produced:

$$\mathbf{u} = E_T(t) \in \mathbb{R}^D$$

The similarity between two vectors is calculated as the cosine of the angle between them:

$$\text{cosine\_similarity}(u, v) = \frac{u \cdot v}{\|u\|\|v\|}$$

- $u \cdot v$ is the dot product of vectors u and v
- $\|u\|$ and $\|v\|$ are the magnitudes of vectors u and v respectively.

4.1.2 Baseline Approach with CLIP

The exceptional capacity of the CLIP model for open-text keyword matching stems directly from its training on the proprietary WebImageText (WIT) dataset, which comprises 400 million image-caption pairs. This massive, diverse, and natural-language dataset allowed the model to map images and arbitrary text concepts into a highly semantic, shared embedding space. Unlike models constrained by a fixed set of training categories, CLIP's exposure to free-form captions enabled it to develop a robust understanding of visual concepts and their linguistic descriptions, thus facilitating powerful generalization. This design permits the model to successfully perform keyword matching and identify visual content corresponding to novel, unseen text queries. The remainder of this section discusses how the CLIP model is used to address multi-keyword queries as a baseline approach.

To efficiently retrieve a set of target images, R, for a query Q, the initial step is to compute vector representations of the entire image repository using the CLIP image encoder during offline pre-processing. These resulting vectors are then stored in an $N \times D$ matrix, where N is the number of vectors and D is the dimensionality. During online query answering, a query Q is treated as a single text string by concatenating its keywords. The concatenated text is then fed into the CLIP text encoder to produce its corresponding representative vector. Then, the images can be sorted by cosine similarity between the vector

of Q and the vector of each image, and the ordered candidate list, X', can be constructed by selecting a specified number of images from the ranked list. The images in X' will be fed into model M in sorted order until k result images are obtained. Algorithm-1 implements the general idea discussed above.

Algorithm 1 Baseline Approach

procedure OFFLINEINDEX(X)
    $N \leftarrow$ Length(X)
    $m \leftarrow$ InitializeZeroMatrix(N, d)
    for $i = 1$ to N **do**
        $v_i \leftarrow= E_I(X[I])$
        $m[i, :] \leftarrow V_i$
    end for
    return m
end procedure
procedure ONLINERETRIEVAL(Q, X, m,k)
    $s_Q \leftarrow$ ConcatenateKeywords(Q)
    $v_Q \leftarrow \mathrm{E}_T(s_Q)$
    $scores \leftarrow v_Q \cdot m$
    $indices \leftarrow$ SortIndicesDescending($scores$)
    $result\_images \leftarrow \emptyset$
    for $i \in indices$ **do**
        $x_i \leftarrow X[i]$
        if $M(x_i)$ is True **then**
            Append($(result\_images, x_i)$)
        end if
        if Length($result\_images$) $= k$ **then**
            break
        end if
    end for
    return $result\_images$
end procedure

While CLIP offers a simple baseline approach, its deficiencies are frequently observed in dense vision applications [31][29]. The CLIP embedding similarity score is effective for establishing an initial candidate pool, but the resulting

deterministic ranking does not inherently guarantee retrieval accuracy. To address this limitation, we replace the rigid ordering mechanism with a stochastic candidate selection process. This novel approach leverages image clustering algorithms to define distinct partitions within the embedding space, thereby enabling the sampling of prospective candidates from diverse clusters.

4.2 Proposed Method

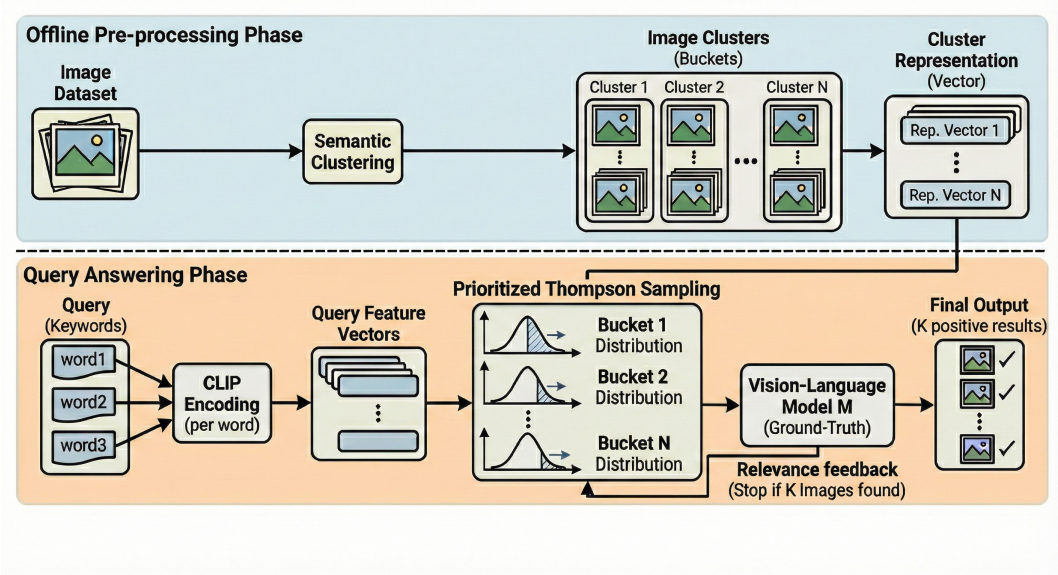

Figure 4.1: Prioritized Thompson Sampling Image Retrieval(PTSIR) Overview

The proposed method, Prioritized Thompson Sampling Image Retrieval, is a two-stage framework. The offline preprocessing phase begins by applying semantic clustering to the image dataset, thereby grouping content-similar images. A cluster representation is then generated for each partition, typically by deriving a representative vector. During the query answering phase, the

query keywords are first encoded into feature vectors using the CLIP model. Subsequently, we sort candidate images within each cluster by their similarity to the query vector set, and we employ a modified Thompson Sampling in which images with higher similarity scores are more likely to be drawn from the selected cluster. These selected images are then sequentially submitted to the high-fidelity model M for matching verification until k positive results are obtained.

The subsequent sections systematically detail the components of this framework. Section-4.2.1 introduces our approach to clustering images. Building on the image clusters, Section 4.2.2 outlines the Image Retrieval by Thompson Sampling methodology, demonstrating how probabilistic sampling techniques can be leveraged to optimize image retrieval. Finally, section-4.3 proposed the Image Tailing as an optimization measure designed to capture more local and detailed information from images to mitigate the deficiency inherited from CLIP.

4.2.1 Image Clustering

This section describes the methodology for clustering images into discrete clusters. The initial step involves transforming image data into high-dimensional vectors using CLIP. We assume that images sharing similar keywords exhibit semantic similarity. The resultant vectors are then grouped using a clustering algorithm to group semantically identical images into the same clusters.

The primary objective of this pre-processing step is to transform the clusters into stochastic buckets of candidate target images for retrieval. By sampling

images from these clusters rather than relying on a fixed, deterministic order, randomness is intentionally introduced into the retrieval process, thereby helping to overcome the limitations of purely deterministic ranking methods.

Because the embedding vectors are high-dimensional (e.g., 512 for CLIP), clustering becomes difficult due to the curse of dimensionality. Therefore, an efficient dimensionality-reduction algorithm is applied to reduce vector dimensionality. UMAP[14] is applied as it has been shown to preserve more of the local and global features of high-dimensional data in lower projected dimensions. The UMAP can be formally defined as a function:

$$UMAP : (\mathbf{X_{UMAP}}, \theta_{\mathbf{UMAP}}) \mapsto \mathbf{Y}$$

$\mathbf{X_{UMAP}}$ is the high-dimensional dataset, $\mathbf{X_{UMAP}} \in \mathbb{R}^{N' \times D'}$ in which N' is the number of data points and D' is the dimension of the data point. $\theta_{\mathbf{UMAP}}$ is the hyperparameter controlling dimension reduction. The resultant vector sets, $\mathbf{Y}$, represent low-dimensional data points after reduction, $\mathbf{Y} \in \mathbb{R}^{\ N' \times d'}$, in which N' is the number of data points and d' is the dimension of the data point.

The reduced embeddings are clustered using HDBSCAN [13]. It is a powerful and modern clustering algorithm that extends the principles of the well-known DBSCAN algorithm. Like DBSCAN, it is a density-based method, meaning it groups data points that are densely sampled. HDBSCAN models clusters using a soft-clustering approach, allowing noise to be modelled as outliers. This prevents unrelated documents from being assigned to any cluster. HDBSCAN can be defined as a function that transforms a dataset $\mathbf{I_{HDBSCAN}}$ into a final

set of stable clusters $\mathbf{L}$:

$$HDBSCAN : (\mathbf{X_{HDBSCAN}}, \theta_{\mathbf{HDBSCAN}}) \mapsto \mathbf{L}$$

$\mathbf{X_{HDBSCAN}}$ is the dataset to be clustered, $\mathbf{X_{HDBSCAN}} \in \mathbb{R}^{N' \times D'}$ in which N' is the number of data points and D' is the dimension of the data point. $\theta_{\mathbf{UMAP}}$ is the hyperparameter controlling clustering algorithm. $\mathbf{L}$ represents the final set clusters with size of P, $\mathbf{L} = \{C_1, \ldots, C_P, C_{noise}\}$. Images in each cluster are a subset of the input points $\mathbf{X_{HDBSCAN}}$. There is an additional cluster, C_{noise}, representing noise or outliers. Images in C_{noise} refer to the images that do not belong to any cluster.The final $\mathbf{L} = \{C_1, \ldots, C_P, C_{\text{noise}}\}$.

HDBSCAN is a density-based algorithm designed to find clusters of arbitrary, non-convex shapes; therefore, we use the medoid as the representative vector for each cluster. The medoid is the actual data point within a cluster that has the minimum total distance to all other points in that cluster. Given a cluster C, the medoid is calculated as:

$$\mathbf{e}_{\text{medoid}} = \underset{\mathbf{e}_i \in C}{\operatorname{argmin}} \sum_{\mathbf{e}_j \in C} cosine\_similarity(\mathbf{e}_i, \mathbf{e}_j)$$

The image clustering approach outlined above is realized by Algorithm-2. The algorithm yields a set of image clusters, each accompanied by its corresponding representative vector.

Algorithm 2 Image Clustering Algorithm

```
1: procedure ImageClustering(X)
2:     V ← E_T(X)
3:     V_reduced ← UMAP(V)
4:     L ← HDBSCAN(V_reduced)
5:     medoids ← ∅
6:     for k = 1 to Length(L) do
7:         medoids_k ← FindMedoid(C_k)
8:     end for
9:     return L, medoids
10: end procedure
```

4.2.2 Image Selection with Thompson Sampling

The primary objective of this section is to elaborate on how sampling techniques can be employed to select images from the generated cluster described in the previous section. section-4.2.2.1 describes the Multi-armed Bandit Problem(MAB) and how image retrieval is generalized as a MAB problem. In section-4.2.2.2, Thompson Sampling is introduced as a standard solution for MAB problems, and some modifications are added onto the standard Thompson Sampling to solve our limit-k image retrieval problem better.

4.2.2.1 Image Retrieval as a Multi-armed Bandit Problem

Following image clustering, a sampling strategy is required to select images from each cluster for verification by the model M. To address this selection challenge, we formulate this process as a Multi-Armed Bandit (MAB) problem. We begin by briefly introducing the fundamental concepts of the MAB problem.

The multi-armed bandit (MAB) can be conceptualized as a collection of K real distributions, $Z = \{B_1, \ldots, B_K\}$. Each distribution in this set represents

the rewards delivered by one of the K available arms. Let $\{\mu_1 \dots \mu_K\}$ be the mean values associated with these reward distributions, and the optimal expected reward is defined as $\mu^* = \max_i \mu_i$, indicating the theoretically optimal reward that could be obtained. The process involves repeatedly selecting an arm, observing the reward, and deciding which arms to choose. The objective is to maximize the cumulative reward.

In the context of limit-k image retrieval, image clusters can be conceptualized as arms within a multi-armed bandit framework. Each cluster is expected to have its own reward distribution based on the query Q. The action involves selecting a specific cluster to identify a subsequent candidate image. The output of model M determines the reward associated with this action. However, our objective has been redefined to minimize the number of time steps required to achieve a cumulative reward of k.

Regret is a metric for evaluating the performance of the agent's strategy. It measures the difference between the total reward the agent actually received and the total reward they could have received if they had perfect advance knowledge of which arm was the best one to pull throughout the entire process. If the agent knew μ^* beforehand, the optimal strategy would be to pull that best arm in every single round t, and the reward for that around is $\hat{r}_t$, so the expected cumulative regret, p, is calculated as:

$$p = \sum_{t=1}^{T} \mu^* - \hat{r}_t$$

The rate at which cumulative regret increases with the number of trials

serves as the primary metric for assessing the algorithmic efficiency in solving the MAB problem. A good algorithm should provide a theoretical guarantee on how quickly this regret grows as the number of trials increases.

The MAB problem inherently involves the Exploration versus Exploitation dilemma. At each step t, the agent must decide whether to exploit the arm with the highest estimated mean reward to maximize immediate gain, or to explore an arm that has been chosen less often to gather more information that could lead to a higher long-term cumulative reward. A solution policy aims to strike an optimal balance between these two conflicting objectives, thereby minimizing regret while maximizing reward.

4.2.2.2 Image Selection via Thompson Sampling

Thompson sampling (TS) [18] is an MAB algorithm that sequentially selects actions to maximize cumulative rewards. Thompson Sampling employs a Bayesian approach. Instead of tracking point estimates for μ_i, it maintains a probability distribution over the possible values of the parameters θ_i, which are updated over time.

Let $\mathcal{H}_t$ denote the history of observations up to time t, including all previous arm pulls and their resulting rewards:

$$\mathcal{H}_t = \{(B_1, \hat{r}_1), (B_2, \hat{r}_2), \ldots, (B_t, \hat{r}_t)\}$$

The agent maintains a posterior probability distribution for the parameter

θ_b of each arm b:

$$P(\theta_b|\mathcal{H}_t)$$

This distribution reflects the agent's current belief about the true value of the arm's parameter θ_b, based on the evidence collected to date.

In this paper, we consider the Bernoulli bandit setting, where rewards for each action are binary (0 or 1), and we use the Beta distribution as the prior. The prior is initialized as Beta$(1, 1)$, because it represents a uniform prior, which means the algorithm starts with maximum uncertainty and no bias regarding the true reward rate of any arm. Since Beta$(1, 1)$ is equivalent to a uniform distribution with a neutral expected value of 0.5, it provides a balanced and mathematically convenient starting point. Let S_b be the number of successes (1s) and F_b be the number of failures (0s) for arm b. In the limit-k image retrieval context, S_b indicates the number of images that model M returns true for verification of matching with query Q, and F_b indicates the number of images that failed the verification. After observing S_b successes and F_b failures, the posterior distribution is:

$$P(\mu_b|\mathcal{H}_t) = \text{Beta}(1 + S_b, 1 + F_b)$$

At time t, the agent samples a success probability from the posterior for each arm, $\tilde{\mu}_{b,t} \sim \text{Beta}(1 + S_b, 1 + F_b)$, and choose the arm $B_t = \arg\max_b \tilde{\mu}_{b,t}$. It has been proven that Thompson Sampling achieves the asymptotically optimal logarithmic regret bound, $O(\log T)$[18].

To apply Thompson Sampling to limit-k image retrieval for a query Q,

all clusters are initially assigned a neutral prior distribution, Beta(1, 1). In each iteration, the algorithm samples a success probability from the current Beta posterior distribution for every cluster k. The cluster with the highest sampled success probability is selected for the next image draw, ensuring that the system exploits clusters currently estimated to be highly rewarding while also exploring less-sampled clusters, given the inherent uncertainty in their broad Beta distribution. An image is then randomly sampled from the chosen cluster and fed to the model M for verification. If the verification is successful, the parameters of the cluster's Beta distribution are updated by incrementing the success count ($S_b \leftarrow S_b + 1$). If it fails, the failure count is incremented ($F_b \leftarrow F_b + 1$). This iterative process continues until k target images are retrieved.

However, reward scarcity is an inherent issue for standard Thompson Sampling in the limit-k image retrieval setting. The reward-scarcity issue arises in contexts in which obtaining a reward signal is infrequent relative to the number of actions taken. This scarcity can severely degrade the performance of standard TS, which relies on observed reward signals to accurately update the posterior distribution of expected rewards. In limited-k image retrieval, the target images are highly scarce in the image repository and poorly represented even within individual clusters. Therefore, it is suboptimal for our objectives to retrieve k target images in the fewest steps when an image is randomly selected from the cluster at time t. Consequently, it is imperative to prioritize selecting clusters and candidates that have higher probabilities of yielding a reward. To address this, we add an enhancement to select clusters and candidates in the

sampling process.

4.2.2.3 Prioritized Thompson Sampling

In this section, we will discuss our strategy for prioritizing the selection of clusters and images that are more semantically similar to the query, Q, as well as our method for selecting a batch of pictures from a cluster at time t rather than a single candidate image.

First, clusters whose representative vectors are closer to the vector of query, Q, are initialized with extra success reward, for example, as Beta$(2, 1)$. This shifts the initial uniform distribution toward a higher probability of success, so these clusters are more likely to be selected initially.

Second, after a cluster is selected, the candidate images within it are ranked by cosine similarity between their embeddings and the textual query. Images are then chosen according to this ranking, which aims to promote promising candidates for priority selection. While cosine similarity does not provide an exact ranking, it provides general guidance, as discussed in Section-4.1.2.

Third, we propose that selecting a batch of images from a prioritized cluster in each Thompson Sampling (TS) round effectively mitigates reward scarcity by increasing exploitation efficiency. In data-scarce scenarios, such as limited-k image retrieval, single-candidate sampling often yields long sequences of zero rewards, which stall the convergence of the arm's posterior distribution. By evaluating a batch simultaneously, the system increases the number of candidates tested per pull, thereby significantly increasing the probability of observing a positive outcome. This rapid feedback reduces uncertainty in the

cluster's posterior, enabling more accurate sampling and a faster transition from exploration to exploitation, thereby overcoming stagnation caused by sparse reward signals.

Algorithm-3 describes the prioritized Thompson Sampling for image retrieval. It is assumed that image clustering of the repository X is completed as a preprocessing step, yielding the set of clusters L before Thompson Sampling is applied for image retrieval. The algorithm first initializes Beta distribution priors for each cluster. Then, the images in each cluster are sorted by their cosine similarity to query Q. The resultant L' represents a set of clusters in which images are already sorted.

The sampling process begins by computing the success probability for each cluster using its Beta distribution, and selecting the cluster with the highest success probability as p_t. An image x_t is then taken from the selected cluster $L'[p_t]$ based on the ranking order in the cluster, and its reward to the query Q is assessed by the function M. Based on the verification result, the corresponding success count, S_{p_t} and failure count, F_{p_t}, are updated for the cluster. The sampling process keeps running until k target images are retrieved.

The algorithm will return the retrieved k images.

Since we proposed to prioritize the selection of clusters and images based on the cosine similarity between vectors generated by CLIP, this guidance will suffer from the deficiency of fine-grained information as well, as mentioned in 4.1.2. In the next section, we propose a simple yet effective approach to mitigate this deficiency for better guidance in the sampling process.

Algorithm 3 Run Prioritized Clustered TS Trial

procedure PRIORITIZEDTHOMPSONSAMPLING(X, Q, L, k)
  $S, F \leftarrow$ InitializePrioritizedPriors($Q, L, mediods$)
  $L' \leftarrow$ SortImagesInCluster(Q, L)
  $P \leftarrow$ Length(L')
  $result\_images \leftarrow \emptyset$
  for $i = 1$ to P **do**
    $\Theta_i \sim$ Beta(S_i, F_i)
  end for
  for Length($result\_images$) $< k$ **do**
    $p_t \leftarrow \arg\max_{p \in \{1,\ldots,P\}} \Theta_p$
    $images\_order \leftarrow L'[p_t]$
    $batch\_candidates \leftarrow$ Next batch of candidates in $images\_order$
    for x_c in $batch\_candidates$ **do**
      if $M(x_c, Q) = 1$ **then**
        $S_p \leftarrow S_p + 1$
        Append($result\_images, x_c$)
      else
        $F_p \leftarrow F_p + 1$
      end if
    end for
  end for
  return $result\_images$
end procedure

4.3 Image Tiling

To optimize retrieval efficiency, we propose image tiling to address the limitations of CLIP's global embeddings, which lack sensitivity to local features. Our method enhances performance by segmenting images into tiles and processing each tile with the CLIP encoder, generating embeddings that capture local visual features. We proposed two algorithms for image tiling: Fixed Uniform Tiling and Segmentation-Guided Tiling.

4.3.1 Fixed Uniform Tiling

Fixed Uniform Tiling is a straightforward image decomposition strategy that partitions an image into a predefined, fixed grid (e.g., 3×3 or 4×4), disregarding the image's semantic content. This method is highly efficient, requiring only simple arithmetic operations on pixel dimensions to calculate the bounding boxes, thus imposing minimal computational overhead. However, its rigid, arbitrary boundaries present a significant drawback. When

4.3.2 Segmentation-Guided Tiling

Image segmentation is a core computer vision (CV) technique that partitions a digital image into multiple pixel subgroups, or image segments. Its primary objective is to simplify the image's representation, making it more meaningful and easier to analyze. Unlike object detection, which only localizes objects with a rectangular bounding box, segmentation provides a precise pixel-level map of an object's shape. This process assigns a label to every pixel, enabling AI models to understand the exact boundaries of entities within a scene.

However, pixel-level boundaries are inherently non-regular, which complicates their direct use for image cropping. To facilitate the cropping process, we approximate these boundaries with a minimum bounding rectangle that encloses all pixels in the original segmentation mask. Segmentation-guided tiling involves running a dedicated image segmentation model on the input image x. The segmentation model takes an image as input and outputs transformed

minimum bounding boxes of size B. The segmentation model is defined as:

$$Seg : (image) \mapsto BBox = \{(x_{\min}, y_{\min}, x_{\max}, y_{\max})\}^B$$

There are two issues associated with the segmentation-guided tiling mentioned above. First, the segmentation model typically generates a large number of bounding boxes as results. Processing and storing each of these boxes presents significant computational and storage challenges. Second, some bounding boxes are too small to crop, resulting in low dimensions that degrade the accuracy of the embedding vectors generated by Clip[22]. Therefore, the resulting bounding boxes are clustered using k-means around their centroids to define spatially meaningful regions. The final tiles are constructed by merging the bounding boxes within each cluster into a new minimum bounding box that encloses all segment bounding boxes in that cluster. This method ensures that the extracted feature vector of a tile contains enough local features while keeping the number of tiles low.

The algorithm-4 implements the segmentation-guided tiling discussed above. The number of clusters (k) in the K-means algorithm is heuristically set to the square root of the total number of bounding boxes. This algorithm returns a list of cropped images.

After defining an image tiling method for a single image, the image tiling algorithm for the entire image repository X is described in algorithm-5. The ImageRepositoryTiling procedure takes an image repository X as input and systematically decomposes each image in X into a set of smaller, distinct image

Algorithm 4 Segmentation-Guided Tiling

procedure SEGMODELCROPIMAGE(x)
    $bboxes \leftarrow$ SegModel(x)
    $center \leftarrow \emptyset$
    for box in $bboxes$ **do**
        $(x, y, w, h) \leftarrow$ ExtractCoordinates(box)
        $centers$.Append($[x + w/2, y + h/2]$)
    end for
    if Length(centers) $= 0$ **then**
        return $[x]$
    end if
    $c \leftarrow \lceil \sqrt{\text{Length}(centers)} \rceil$
    $labels \leftarrow KMeansCluster(centers, n\_clusters = c)$
    $fbox \leftarrow$ EmptyDictionary
    for idx from 1 to Length($labels$) **do**
        $cluster\_id \leftarrow labels[idx]$
        $coordinates \leftarrow$ ExtractBBoxCoordinates($bboxes[idx]$)
        $fbox[cluster\_id]$.Append($coordinates$)
    end for
    $cropped\_imgs \leftarrow \emptyset$
    for $cluster\_id$ in $fbox.keys$ **do**
        $merged\_box \leftarrow$ MergeRectangles($fbox[cluster\_id]$)
        $cropped\_img \leftarrow$ image.Crop($merged\_box$)
        $cropped\_imgs$.Append($cropped\_box$)
    end for
    return $cropped\_imgs$
end procedure

tiles. The method returns two key outputs: the complete set of generated image tiles, X_{tiles}, and a mapping that explicitly records the parent-child relationship between each original image and its corresponding derived tiles, $m_{original\_tiles}$. The ImageTiling algorithm could be replaced with either Fixed Uniform Tiling or Segmentation-Guided Tiling. We will compare them in experiments.

Image tiling divides the original image into smaller patches, each representing local features. In 4.1.1, the similarity between a query Q and an image x

Algorithm 5 Algorithm for Image Tiling

procedure ImageRepositoryTiling(X)

    $X_{tiles} \leftarrow \emptyset$
    $m_{original\_tiles} \leftarrow$ Map()
    for each x_i in X **do**
        $x_{i\_tiles} \leftarrow$ ImageTiling(i_k)
        $X_{tiles} \leftarrow X_{tiles} \cup \{X_{i\_tiles}\}$
        $m_{original\_tiles}[\text{ID}(x_i)] = x_{i\_tiles}$
    end for
    Return X_{tiles}, $m_{original\_tiles}$
end procedure

is calculated as cosine similarity between their embedding vectors. However, this formula cannot be applied directly after image tiling because an image is represented as a set of embedding vectors rather than a single vector. Therefore, in the next section, we introduce a new method of calculating similarity between a query and an image.

4.3.2.1 Similarity Calculation After Image Tiling

The image tiling process decomposes an image into several smaller pieces, with each piece generating an embedding vector; thus, an image is represented as a set of embedding vectors. Similarly, instead of concatenating keywords, the textual query is defined as a set of embedding vectors, where each vector is derived from a single keyword. Thus, we transitioned from cosine similarity between two single global vectors to a method that computes the similarity between the two resulting sets of localized embedding vectors. In the following part of this section, we discuss how to apply the $MaxSim$ operation to calculate the similarity between two sets of vectors.

The $MaxSim$ is a similarity aggregation technique commonly used in embedding-based matching tasks. Instead of comparing two items using a single global similarity score, $MaxSim$ computes many pairwise similarities between their constituent representations, and then takes the sum of the maximum among these values. This approach captures the strongest local correspondence between two entities, enabling finer-grained, context-aware matching. $MaxSim$ emphasizes discriminative features and is robust to irrelevant or noisy components in the representation. Formally, the similarity between a query Q and an image x, and x_j represents the vector of the j_{th} tile of the original image x.

$$S(Q, x) = \sum_{i=1}^{m} \max_{1 \leq j \leq n} \text{cosine\_similarity}(E_T(q_i), E_I(x_j))$$

The image tiling mechanism is introduced after the initial image clustering. Each image in the repository is decomposed into constituent tiles, with a tiling map recording the parent-child relationship between the original image and its tiles. The tile embeddings are then computed via the CLIP image encoder and stored. During query response, the simple cosine similarity used by the SortImagesInCluster function (Algorithm 3) is replaced with a novel tile-based similarity measure. Apart from this substitution, the subsequent steps of the retrieval algorithm are preserved.

Chapter 5

Experiments

5.1 Experiment Setup

5.1.1 Experiment Platform

The experimental environment was hosted on Google Colab. The system was equipped with an NVIDIA A100 GPU featuring 80GB of VRAM, enabling the efficient processing of large-scale models and datasets. This GPU was integrated into a hardware configuration that included 167GB of system RAM and 235GB of disk storage. The software environment ran on Python 3.10+.

5.1.2 Dataset

We conducted our experiments on the Microsoft COCO (Common Objects in Context) dataset [11]. COCO is a large-scale, widely recognized benchmark for object detection, segmentation, and captioning tasks. It contains images including multiple objects in a natural context. The 2017 version of the dataset,

which we used, contains over 118,000 images for training and 5,000 images for validation. The COCO dataset comprises annotations across 80 object categories, including "person," "car," and "dog." The high density of objects in each image makes COCO an ideal yet challenging benchmark for assessing the performance of our method in retrieving images for multi-keyword queries.

5.1.3 Query

Queries were synthesized by randomly selecting two to five category names from the COCO dataset. This procedure simulates a realistic open-text scenario, as our method operates without prior knowledge of these categories. In total, 500 queries were generated, each guaranteed to yield at least 20 result images.

5.1.4 Models

In addition to the CLIP model discussed in 4.1.1, our method incorporates several other models, which are introduced in this section.

We use BLIP[10] as the ground-truth model, M, for the verification of matching between query Q and an image x. BLIP (Bootstrapping Language–Image Pre-training) is a multi-modal framework designed to align visual and textual representations for tasks such as retrieval, captioning, and VQA. It combines separate image and text encoders with a multimodal transformer that integrates information via cross-modal attention, in which queries from one modality attend to key–value pairs from the other. This mechanism enables BLIP to highlight relevant image regions for specific words and to refine textual understanding based on visual context, thereby enabling fine-grained alignment

beyond simple feature fusion.

The segmentation model used in section-4.2.1 is based on the YOLO11[8]. YOLO11 is an advanced real-time vision model that extends the YOLO family [17] with stronger feature extraction and multi-scale reasoning. Its instance segmentation head predicts both object locations and precise pixel-level masks, allowing it to distinguish overlapping objects and capture fine-grained shapes. This combination of speed and detailed segmentation makes YOLO11 highly effective for real-world, latency-sensitive applications.

5.1.5 Methods

Our experiments are designed to evaluate the performance of the proposed Prioritized Thompson Sampling Image Retrieval(PTSIR) method. We achieve this by comparing its results with those of the baseline approach and variants of the proposed PTSIR method.

To comprehensively evaluate our approach, we employed three distinct methods: the Baseline approach algorithm-1, the proposed Prioritized Thompson Sampling Image Retrieval(PTSIR) method with image tiling optimization, which is default as segmentation-guided tiling, and a variant PTSIR-NoTiling, which excludes the optimization of image tiling as discussed in section-4.3.

The PTSIR and PTSIR-NoTiling shared prioritization settings introduced in section-4.2.2.3. The default sampling batch size is 1, and the percentage of the prioritized cluster is 15%. They both employed the candidates' prioritization, but used different ranking metrics. PTSIR ranked candidates in clusters based on the image tiling method, while PTSIR-Notiling ranked candidates in clusters

based on the naive cosine similarity between CLIP embeddings.

5.2 Experiment Result

We quantify retrieval efficiency in this section by systematically measuring the average number of calls to model M required to retrieve k target images, with k ranging from 1 to 12. The analysis is structured to first benchmark the proposed PTSIR method against the baseline approach and the PTSIR-NoTiling variant. To comprehensively analyze the proposed PTSIR model, we also compared its performance against standard Thompson Sampling in image retrieval. We then performed an ablation study by adjusting PTSIR's parameters to evaluate its behaviour and robustness across various configurations.

5.2.1 Comparison Between Methods

The experiment results are plotted with the x-axis representing the number of target images (k) and the y-axis representing the average number of calls to Model M. Figure-5.1 compares the efficiency of the baseline method, PTSIR, and PTSIR-NoTiling.

As shown in Figure-5.1, the proposed PTSIR and PTSIR-NoTiling methods exhibit superior performance compared to the baseline approach, particularly as the number of requested target images (k) increases. When k is small, PTSIR and PTSIR-NoTiling initially require more calls to model M. This is because the Thompson sampling approach necessitates a few rounds of exploration to identify clusters likely to contain target images, based on feedback from M.

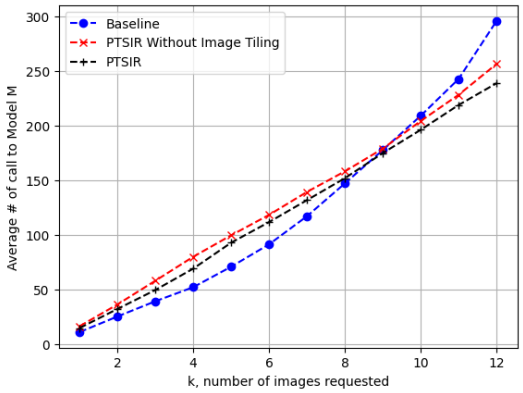

Figure 5.1: Comparison Between Baseline, PTSIR, and PTSIR-NoTiling

However, the baseline approach shows an exponential increase in model M calls as k rises. In contrast, PTSIR and PTSIR-NoTiling maintain a linear relationship between k and the number of calls. This fundamental difference in scaling demonstrates that PTSIR and PTSIR-NoTiling significantly lower the growth rate of calls to model M from exponential to linear as k increases, confirming its superior efficiency at high k values. The superiority of PTSIR and PTSIR-NoTiling is attributed to the inherited guarantee provided by the Thompson Sampling on how quickly this regret grows as the number of trials increases, as discussed in section-4.2.2.2.

The performance gap between PTSIR and the ablation variant PTSIR-NoTiling is marginal. We initially hypothesized that the image tiling technique would yield a more substantial improvement by providing better guidance to the sampling process, given its theoretical ability to capture more local and detailed image features. The minimal observed difference warrants further investigation into the interaction between tiling and the candidate selection logic. We assess the utility of image tiling by measuring its effectiveness in

improving the identification of candidate target images compared with a simple CLIP embedding. This analysis compares Segmentation-Guided Tiling (SGT) with Fixed Uniform Tiling (FUT).

5.2.2 Image Tiling

In this section, we investigated the contribution of image tiling and the associated $MaxSim$ operation to enhancing image ranking accuracy. This was achieved by evaluating the mAP@K metric, comparing the ranking performance of a single global embedding with that of tiled embeddings. The analysis included two distinct tiling strategies: Fixed Uniform Tiling (FUT) and Segmentation-Guided Tiling (SGT).

Table 5.1: mAP comparison between ranking by CLIP, FUT and SGT

Method	**mAP@5**	**mAP@10**	**mAP@15**	**mAP@25**	**mAP@50**
CLIP	0.42	0.40	0.39	0.36	0.31
FUT	0.43	0.42	0.39	0.37	0.33
SGT	**0.51**	**0.49**	**0.45**	**0.42**	**0.36**

The ranking of candidates based on Segmentation-Guided Tiling achieves the highest mAP across all ranges. Fixed Uniform Tiling performed slightly better than CLIP but worse than Segmentation-Guide Tiling. Image tiling improves predictive performance, resulting in better prioritization and placement of candidate target images at the highest ranks. However, this improvement in deterministic order does not significantly affect Thompson Sampling, as it is a random process that is not constrained by the deterministic ranking.

The subsequent analysis examines three options we applied in PTSIR:

prioritized cluster selection, prioritized candidate selection, and batch sampling.

5.2.3 Prioritized Clusters in PTSIR

In this section, we examine how cluster prioritization improves the efficiency of the PTSIR method. We quantified this improvement by measuring the average number of calls to the ground truth model M. For this experiment, the number of requested images, k, was fixed at 10, the sampling batch size was 1, and the percentage of prioritized clusters was gradually increased from 0% to 90%.

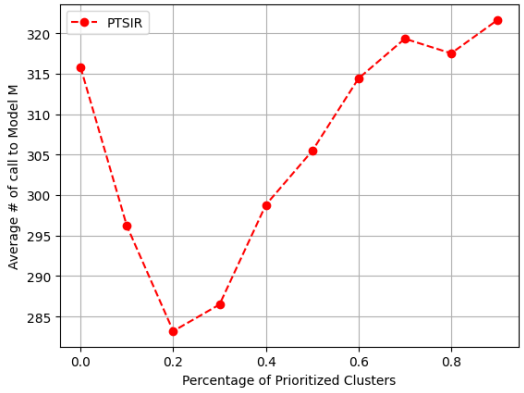

Figure 5.2: Comparison Between Different Percentages of Prioritized Clusters in PTSIR

The figure-5.2 depicts the non-linear relationship between the percentage of prioritized clusters and efficiency for the PTSIR. The plot shows a V-shaped trend, in which increasing the rate of prioritized clusters from 0 to 20% results in a sharp reduction in cost. However, prioritizing beyond the optimal point (approximately 20%) causes the price to rise steadily and converge to the level observed under no prioritization. This demonstrates that prioritizing a small subset of clusters is highly effective at minimizing calls to the ground-truth

model for verification, but excessive prioritization cancels out these gains. This is expected because, as the percentage of prioritized clusters increases, the clusters are no longer effectively distinguished for the specific query Q. Consequently, performance reverts to the level achieved with no prioritization.

5.2.4 Candidates Prioritization in PTSIR

In this section, we evaluate how candidate prioritization improves the efficiency of the PTSIR. For this experiment, we fixed the batch size to 1 and used no prioritized clusters. Without candidate prioritization, the candidate images are randomly selected from each cluster; because the batch size is one and no cluster is prioritized, this is equivalent to the standard Thompson Sampling method. We measured the average number of calls to the ground-truth model M as a function of the number of request images k.

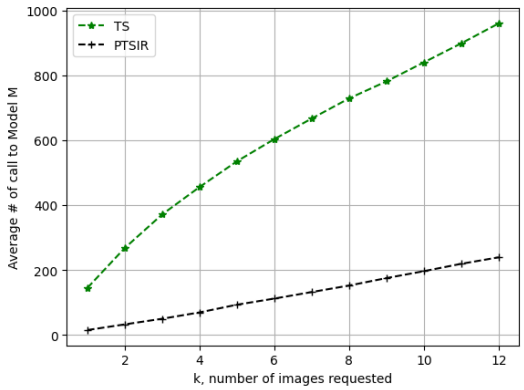

Figure 5.3: Comparison Between Standard Thompson Sampling and Prioritized Thompson Sampling Image Retrieval

In figure-5.3, we refer to the sampling process without candidate prioritization as the standard Thompson Sampling process, shown in green series.

There is a significant performance gap between the standard Thompson Sampling method and our proposed PTSIR method. The performance of standard Thompson Sampling is severely degraded by the scarcity of target images in the repository, as discussed in Section-4.2.2.3. Even within the most relevant image clusters, the low true-positive rate means random sampling rarely selects them. This lack of positive feedback prevents the reward distributions for these clusters from being meaningfully updated, leading to stagnation. Conversely, the proposed PTSIR addresses this issue by employing a prioritization mechanism that guides selection toward candidate target images. While standard TS is ineffective, its observed near-logarithmic growth trend is consistent with its asymptotically optimal logarithmic regret bound.

5.2.5 PTSIR with Batch Sampling

In this section, we evaluate whether batch sampling improves the efficiency of PTSIR as a strategy to mitigate the issue of reward scarcity as discussed in section-4.2.2.3. We quantify this by measuring the average number of calls to the ground-truth model M as a function of the batch size. For this experiment, the total number of requested images (k) was fixed at 10, and no prioritized clusters were used, while the batch size was gradually increased from 1 to 25.

As shown in Figure-5.4, the relationship between batch size and computational cost is observed to be fundamentally volatile and non-linear. The overall trend begins with a high number of calls to the ground-truth model M, then declines as the batch size increases, resulting in an extended trough with fluctuations. The batch size within the range of trough marks the most

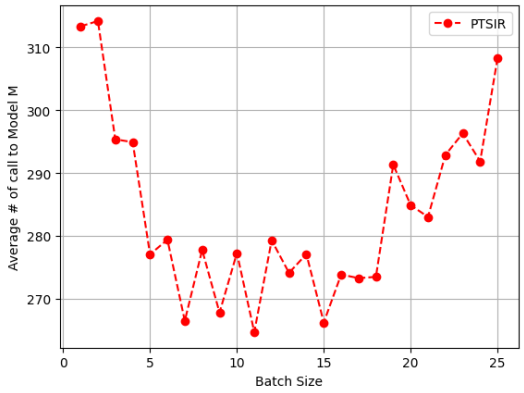

Figure 5.4: Comparison Between Different Sizes of Batch Sampling in PTSIR

efficient system performance. However, as the batch size continues to grow, costs are rising again. The experimental results are expected, as we expect batch sampling to enhance the exploitation of the cluster and to yield a reward within it. The moderate batch size will optimize the sampling process by accelerating feedback, thereby quickly refining the cluster's reward posterior. Consequently, the model achieves more accurate posterior sampling and a faster shift from inefficient exploration to effective exploitation. However, when the batch size becomes too large, the sampling process exhibits exploitation. The large batch forces the system to evaluate many more images than necessary, leading to unnecessary calls to the expensive ground-truth model M.

5.2.6 Evaluation on Number of Clusters

In this section, we evaluate how the number of clusters influences the performance of PTSIR. As noted in previous sections, we employed HDBSCAN as our clustering algorithm because its ability to adapt to local density variations yields more meaningful, interpretable groupings and significantly improved

separation of true semantic clusters. Therefore, it is particularly effective for clustering CLIP embeddings, which often form complex, non-spherical semantic manifolds in high-dimensional space.

HDBSCAN doesn't support explicitly specifying the number of clusters; instead, it uses min_cluster_size to control the number of clusters produced. min_cluster_size is arguably the most critical parameter in HDBSCAN. It controls the minimum cluster size and therefore influences the number of clusters generated. Increasing this value results in fewer, larger clusters, whereas decreasing it produces more clusters.

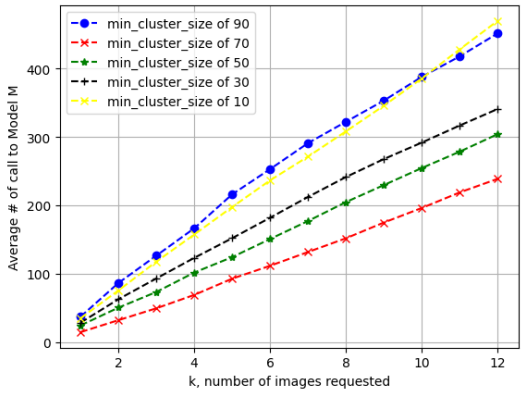

Figure 5.5: Comparison Between Different Sizes of min_cluster_size in clustering

As shown in figure-5.5, the PTSIR achieves the best performance when the min_cluster_size is 70, and the performance starts to drop regardless of whether it increases or decreases. This implies that PTSIR's performance is affected by the number and size of clusters, and the choice of min_cluster_size should be made carefully, as there is an optimal range for this parameter. Based on the experiment, the intuitive number would be the square root of the

repository size.

5.2.7 Time Analysis

In this section, we evaluate the actual processing time between the baseline approach and the proposed PTSIR. Instead of measuring the number of calls to model M, we measured the average answering time per query when k = 12. The answering time is divided into three parts: initialization, sampling, and a call to M.

Initialization time for the baseline approach includes computing the embedding of query Q and sorting the candidates by embedding distance. For PTSIR, in addition to computing the embedding of query Q, it also identifies related topics, initializes the prior distribution, and sorts candidates for each cluster. The sampling time does not apply to the baseline approach. As for PTSIR, it represents the total time spent sampling candidates from the clusters and updating the reward distribution. The verification time refers to the total time spent calling model M for the relevance per query.

Table 5.2: Average execution time analysis, k = 12

Method	**Initialization**	**Sampling**	**Verification**	**Total**
PTSIR	0.67s	0.72s	133.28s	134.67s
Baseline	0.12s	N/A	165.2s	165.32s

Table 5.2 shows that the proposed method incurs overhead during initialization and sampling. Still, this time is negligible compared to the total time used by the model M. Because the PTSIR uses fewer visits to model M, it answers queries more quickly on average.

Chapter 6

Conclusion

This thesis presents Prioritized Thompson Sampling Image Retrieval (PTSIR), a hybrid framework designed to address the efficiency bottlenecks inherent in large-scale, multi-keyword image retrieval. While foundation models like CLIP provide robust global semantic alignment, they lack the fine-grained information required for multi-keyword queries, necessitating the use of expensive verification ground truth oracles. Our research demonstrates that modelling retrieval as a Multi-Armed Bandit (MAB) problem successfully minimizes the computational cost of this verification process.

The most critical finding from our experiments is the shift in cost scaling. The baseline CLIP approach exhibits an exponential increase in calls to the ground-truth model as the number of required target images k increases. In contrast, PTSIR maintains a linear relationship between k and the number of calls to the ground-truth model. This suggests that the probabilistic balancing of exploration and exploitation is significantly more scalable than deterministic

ranking for identifying rare targets in large datasets. By treating image clusters as "arms," the system avoids the redundancy of exhaustively checking high-similarity candidates that dominate the top of deterministic lists.

However, PTSIR performs worse than the baseline when the number of required images k is low because the algorithm incurs an initial exploration overhead to identify which clusters contain relevant target images. While the baseline approach immediately exploits a deterministic ranking to retrieve the highest-similarity candidates first, PTSIR employs a stochastic sampling framework that requires several preliminary rounds to collect feedback and refine its posterior probabilities. Consequently, the baseline can efficiently identify target images at the top of the list at minimal cost. In contrast, PTSIR must incur a setup cost to locate the high-yield clusters before it can efficiently exploit them.

Our comparison with standard Thompson Sampling provides a fundamental insight into the application of MABs in information retrieval. Although standard Thompson Sampling exhibits a linear relationship between k and the number of calls to the ground-truth model, it failed to achieve optimal performance due to reward sparsity, indicating that the target images were too sparse to provide the frequent feedback required to update posterior distributions effectively. This leads to the judgment that some heuristics for possible target images are not merely an optimization, but a necessity for this domain. Therefore, we proposed two measures as heuristics to prioritize the selection of clusters and candidates. Additionally, we suggest using batch sampling to mitigate the reward scarcity issue further.

The experimental data confirm that prioritization in clusters and candidates' images, as well as batch sampling, could enhance the efficiency of PTSIR. Prioritization in candidate image selection contributes to the substantial performance gap between standard Thompson Sampling and PTSIR, as shown in Figure ref fig:stPTS. Our analysis of batch sampling and cluster prioritization reveals a V-shaped performance curve, indicating that the system is sensitive to hyperparameters. In our experimental settings, prioritizing approximately 20% of clusters yielded optimal results; exceeding this threshold degraded performance to baseline levels, as the system lost its ability to distinguish between promising and irrelevant clusters. Moderate batch sizes accelerated posterior updates and improved convergence. However, large batches led to over-exploitation, wasting resources on verifying valid but unnecessary candidates.

A nuanced finding emerges from the ablation study regarding Image Tiling. While Segmentation-Guided Tiling (SGT) achieved the highest Mean Average Precision (mAP) by capturing local features—surpassing both CLIP and Fixed Uniform Tiling—its impact on the overall retrieval efficiency of PTSIR was marginal compared to the non-tiled variant. This leads to the conclusion that while tiling improves the quality of the initial embedding representation, retrieval efficiency is primarily driven by the robust decision-making of the Prioritized Thompson Sampling algorithm. The coarse guidance provided by standard CLIP embeddings, when coupled with intelligent sampling, is sufficient to guide the system, rendering the computationally heavier tiling process optional for efficiency-focused applications.

In summary, PTSIR bridges the gap between fast, coarse-grained search

and slow, fine-grained verification. It proves that in resource-constrained environments, intelligent probabilistic sampling is a viable alternative to exhaustive ranking, provided that the "cold start" problem of reward scarcity is mitigated through prioritization.

Bibliography

[1] Kumail Alhamoud et al. *Vision-Language Models Do Not Understand Negation*. 2025. arXiv: 2501.09425 [cs.CV]. URL: https://arxiv.org/abs/2501.09425.

[2] Declan Campbell et al. "Understanding the limits of vision language models through the lens of the binding problem". In: *Proceedings of the 38th International Conference on Neural Information Processing Systems*. NIPS '24. Vancouver, BC, Canada: Curran Associates Inc., 2024. ISBN: 9798331314385.

[3] Bingyi Cao, André Araujo, and Jack Sim. "Unifying Deep Local and Global Features for Image Search". In: *Computer Vision – ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XX*. Glasgow, United Kingdom: Springer-Verlag, 2020, pp. 726–743. ISBN: 978-3-030-58564-8. DOI: 10.1007/978-3-030-58565-5_43. URL: https://doi.org/10.1007/978-3-030-58565-5_43.

[4] Min Cao et al. *Efficient Image-Text Retrieval via Keyword-Guided Pre-Screening*. 2023. arXiv: 2303.07740 [cs.CV]. URL: https://arxiv.org/abs/2303.07740.

[5] Jiankang Deng et al. "ArcFace: Additive Angular Margin Loss for Deep Face Recognition". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 44.10 (Oct. 2022), pp. 5962–5979. ISSN: 1939-3539. DOI: 10.1109/tpami.2021.3087709. URL: http://dx.doi.org/10.1109/TPAMI.2021.3087709.

[6] Alexey Dosovitskiy et al. *An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale*. 2021. arXiv: 2010.11929 [cs.CV]. URL: https://arxiv.org/abs/2010.11929.

[7] Kaiming He et al. *Deep Residual Learning for Image Recognition*. 2015. arXiv: 1512.03385 [cs.CV]. URL: https://arxiv.org/abs/1512.03385.

[8] Rahima Khanam and Muhammad Hussain. *YOLOv11: An Overview of the Key Architectural Enhancements*. 2024. arXiv: 2410.17725 [cs.CV]. URL: https://arxiv.org/abs/2410.17725.

[9] Martha Lewis et al. "Does CLIP Bind Concepts? Probing Compositionality in Large Image Models". In: *Findings of the Association for Computational Linguistics: EACL 2024*. Ed. by Yvette Graham and Matthew Purver. St. Julian's, Malta: Association for Computational Linguistics, Mar. 2024, pp. 1487–1500. URL: https://aclanthology.org/2024.findings-eacl.101/.

[10] Junnan Li et al. *BLIP: Bootstrapping Language-Image Pre-training for Unified Vision-Language Understanding and Generation*. 2022. arXiv: 2201.12086 [cs.CV]. URL: https://arxiv.org/abs/2201.12086.

[11] Tsung-Yi Lin et al. *Microsoft COCO: Common Objects in Context*. 2015. arXiv: 1405.0312 [cs.CV]. URL: https://arxiv.org/abs/1405.0312.

[12] Haotian Liu et al. *Visual Instruction Tuning*. 2023. arXiv: 2304.08485 [cs.CV]. URL: https://arxiv.org/abs/2304.08485.

[13] Claudia Malzer and Marcus Baum. "A Hybrid Approach To Hierarchical Density-based Cluster Selection". In: *2020 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI)*. IEEE, Sept. 2020, pp. 223–228. DOI: 10.1109/mfi49285.2020.9235263. URL: http://dx.doi.org/10.1109/MFI49285.2020.9235263.

[14] Leland McInnes, John Healy, and James Melville. *UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction*. 2020. arXiv: 1802.03426 [stat.ML]. URL: https://arxiv.org/abs/1802.03426.

[15] Kaleb Newman et al. *Do Pre-trained Vision-Language Models Encode Object States?* 2024. arXiv: 2409.10488 [cs.CV]. URL: https://arxiv.org/abs/2409.10488.

[16] Alec Radford et al. *Learning Transferable Visual Models From Natural Language Supervision*. 2021. arXiv: 2103.00020 [cs.CV]. URL: https://arxiv.org/abs/2103.00020.

[17] Joseph Redmon et al. *You Only Look Once: Unified, Real-Time Object Detection*. 2016. arXiv: 1506.02640 [cs.CV]. URL: https://arxiv.org/abs/1506.02640.

[18] Daniel Russo et al. *A Tutorial on Thompson Sampling*. 2020. arXiv: 1707.02038 [cs.LG]. URL: https://arxiv.org/abs/1707.02038.

[19] Konstantin Schall et al. "Optimizing CLIP Models for Image Retrieval with Maintained Joint-Embedding Alignment". In: *Similarity Search and Applications: 17th International Conference, SISAP 2024, Providence, RI, USA, November 4–6, 2024, Proceedings*. Providence, RI, USA: Springer-Verlag, 2024, pp. 97–110. ISBN: 978-3-031-75822-5. DOI: 10.1007/978-3-031-75823-2_9. URL: https://doi.org/10.1007/978-3-031-75823-2_9.

[20] Jaisidh Singh et al. *Learn "No" to Say "Yes" Better: Improving Vision-Language Models via Negations*. 2024. arXiv: 2403.20312 [cs.CV]. URL: https://arxiv.org/abs/2403.20312.

[21] Hugo Touvron et al. *LLaMA: Open and Efficient Foundation Language Models*. 2023. arXiv: 2302.13971 [cs.CL]. URL: https://arxiv.org/abs/2302.13971.

[22] Weijie Tu, Weijian Deng, and Tom Gedeon. *A Closer Look at the Robustness of Contrastive Language-Image Pre-Training (CLIP)*. 2024. arXiv: 2402.07410 [cs.CV]. URL: https://arxiv.org/abs/2402.07410.

[23] Ashish Vaswani et al. *Attention Is All You Need*. 2023. arXiv: 1706.03762 [cs.CL]. URL: https://arxiv.org/abs/1706.03762.

[24] Min Yang et al. "DOLG: Single-Stage Image Retrieval with Deep Orthogonal Fusion of Local and Global Features". In: *2021 IEEE/CVF*

International Conference on Computer Vision (ICCV). 2021, pp. 11752–11761. DOI: 10.1109/ICCV48922.2021.01156.

[25] Lewei Yao et al. *FILIP: Fine-grained Interactive Language-Image Pre-Training*. 2021. arXiv: 2111.07783 [cs.CV]. URL: https://arxiv.org/abs/2111.07783.

[26] Chun-Hsiao Yeh et al. "Decoupled Contrastive Learning". In: *Computer Vision – ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXVI*. Tel Aviv, Israel: Springer-Verlag, 2022, pp. 668–684. ISBN: 978-3-031-19808-3. DOI: 10.1007/978-3-031-19809-0_38. URL: https://doi.org/10.1007/978-3-031-19809-0_38.

[27] Jiahui Yu et al. *CoCa: Contrastive Captioners are Image-Text Foundation Models*. 2022. arXiv: 2205.01917 [cs.CV]. URL: https://arxiv.org/abs/2205.01917.

[28] Mert Yuksekgonul et al. *When and why vision-language models behave like bags-of-words, and what to do about it?* 2023. arXiv: 2210.01936 [cs.CV]. URL: https://arxiv.org/abs/2210.01936.

[29] Yiwu Zhong et al. *RegionCLIP: Region-based Language-Image Pretraining*. 2021. arXiv: 2112.09106 [cs.CV]. URL: https://arxiv.org/abs/2112.09106.

[30] Yiwu Zhong et al. "RegionCLIP: Region-based Language-Image Pretraining". In: *2022 IEEE/CVF Conference on Computer Vision and Pattern*

Recognition (CVPR). 2022, pp. 16772–16782. DOI: 10.1109/CVPR52688.2022.01629.

[31] Chong Zhou, Chen Change Loy, and Bo Dai. *Extract Free Dense Labels from CLIP*. 2022. arXiv: 2112.01071 [cs.CV]. URL: https://arxiv.org/abs/2112.01071.