PRISM: Privacy-Preserving Semantic Discovery Across Sovereign Image Archives

Kar Balan[1] Tim Wood[2] Junaid Awan[1] Andrew Gilbert[1] John Collomosse[1,3]
[1]DECaDE, University of Surrey [2]Digital Catapult [3]Adobe Research
{k.balan,m.awan,a.gilbert}@surrey.ac.uk tim.wood@digicatapult.org.uk collomos@adobe.com

Abstract

Media provenance standards are increasingly used to express ownership and licensing conditions for images. Yet discovery within sovereign image archives remains largely centralised, often requiring data aggregation or exposure of proprietary collections. We introduce PRISM, a decentralised, privacy-preserving infrastructure for cross-silo semantic discovery via natural-language image search. PRISM enables clients to query distributed image databases without revealing query contents, while archive owners retain confidentiality over their underlying full datasets. By combining vision–language (CLIP) embeddings with homomorphic encryption, approximate semantic similarity is computed across silos without exposing embeddings, intermediate scores, or archive contents—revealing only the items explicitly retrieved to the client. We implement and evaluate PRISM at scale using Recap-DataComp-1B, and demonstrate that encrypted decentralised semantic search achieves accuracy parity with plaintext retrieval with strong confidentiality guarantees.

1. Introduction

Generative AI is raising new concerns over consent and compensation for the re-use of creative content [21]. The ever-increasing demand for AI training data is creating a market for large collection owners to license content, particularly in the cultural heritage sector. Yet in the creative sector, content production is highly decentralised; globally, over 57% of GVA is generated by freelancers or small businesses [31]. There is no easy way for creatives to federate in order to license their content *at scale* in this market.

Emerging solutions to decentralised content licensing extend media provenance standards (*e.g*. C2PA [19]), to express usage rights via further open standards (*e.g*. Open Digital Rights Language [32]) within the metadata of digital assets [4, 5, 20]. These frameworks describe ownership, permissions, and authenticity [6, 21], but they do not address how relevant content is discovered in the first place. Today, discovery at scale largely remains centralised; rights holders, agencies, and repositories must typically upload or mirror content into stock platforms or aggregators to be searchable, trading away data sovereignty and often exposing proprietary collections. This prohibits a creative from entering their content into the decentralised marketplace, for potential remuneration *e.g*. for AI training, without exposing copies of all of their content to the public domain. This is a particular concern given the common practice of scraping publicly visible creative content for AI training.

PRISM addresses this gap, proposing an approach for privacy-preserving semantic discovery across decentralised, independently managed (*i.e*. 'sovereign') image archives. PRISM enables natural-language search over distributed collections whilst protecting both (i) the confidentiality of user queries and (ii) the confidentiality of archive-held content. It functions as an interoperability layer that can sit in front of sovereign image repositories and integrate with downstream provenance and licensing workflows; once an item is explicitly retrieved via PRISM, any authenticity and rights protocols expressed through standards such as C2PA or ODRL may be followed to license it. We make two technical contributions:

1. **Privacy-preserving natural language image search** over large-scale distributed databases. We demonstrate that secure, decentralised search is feasible at scale, combining vision-language models with efficient homomorphic encryption protocols for secure computation.
2. **Permutation protocol** to prevent potential information leakage from repeated queries. Randomly permuting the dataset indices of the embeddings ensures that a client cannot reconstruct the database even when submitting multiple linearly independent queries, preserving the privacy of server-held data while maintaining correctness in the similarity computations.

To the best of our knowledge, PRISM is the first work to integrate these components into a large-scale practical, decentralised, and privacy-preserving search system. While

we demonstrate PRISM for natural-language search over images, the design extends directly to other modalities whenever a shared embedding space exists—for example, CLAP for audio/music [27] and VideoCLIP for video [50]. While we have not proved our system secure in the presence of a quantum adversary, since we use lattice-based homomorphic encryption (and this is the only cryptographic material the adversary observes during protocol execution), our system is also plausibly post-quantum secure.

Together, these contributions address a critical gap in decentralised content ecosystems: enabling fuzzy, semantically meaningful search while preserving the privacy of both user queries and the datasets of content providers. Our approach thus empowers both content producers and consumers, providing a foundation for confidential discovery and fair participation in the decentralised creative economy.

2. Related work

Traditional search systems have relied on keyword or tag-based matching. Platforms such as Shutterstock typically require users to provide textual descriptions or keywords, and retrieval is performed via exact or near-exact matches against this metadata [47]. This approach is limited in two key ways: (1) metadata creation is labour-intensive and subjective, and (2) exact keyword matching fails to capture contextual nuances, synonyms, or semantic relationships critical for richer discovery [36, 37]. While decentralised data marketplaces and personal storage networks (e.g., Ocean Protocol [42], Solid/ESPRESSO [41, 44]) aim to facilitate data sharing and monetisation without central intermediaries, their discovery mechanisms suffer from these same metadata-driven drawbacks. Furthermore, decentralised marketplaces built on public ledgers (e.g., Ethereum NFTs) inherently expose metadata and media, enabling unauthorised scraping and copy-minting [1, 30]. They do not provide the private, semantic discovery layer necessary to protect sovereign creator archives.

Modern information retrieval addresses these limitations using multi-modal models, such as Vision-Language Models (e.g. CLIP [43]), which embed images and text into a shared semantic space. This enables semantic search—retrieval based on approximate or contextual similarity rather than literal token overlap—allowing systems to capture synonyms, paraphrases, or visually related concepts without manual annotation. However, feature embeddings themselves introduce privacy risks, as high-dimensional representations can leak sensitive information [35]. Leaky-CLIP [17] demonstrates that adversaries can reconstruct semantically accurate training images from CLIP embeddings, showing that exchanging embeddings does not inherently guarantee privacy. Thus, to protect proprietary archives and user queries, semantic search must be coupled with privacy-preserving cryptographic techniques.

Existing privacy-preserving image matching systems largely focus on exact or near-duplicate detection rather than semantic search. Breidenbach et al. [10] use perceptual image hashes stored in Bloom filters to enable client-side illegal image re-identification. Queries are hashed but not fully private, as deterministic hashing and the limited hash space can reveal information [48]. Apple's CSAM [2] system employs NeuralHash and a thresholded private set intersection protocol, revealing content only if a sufficient number of matches are found, offering partial query privacy. Other approaches protect query privacy but require the database to remain entirely public [46]. Other decentralised works addressing sovereign visual content archives such as ARCHANGEL [11, 12] focus on media integrity, rather than media discovery. In concurrent work, Zhao [53] noted that Partially (additively) Homomorphic Encryption (PHE) can be used to allow a client to compute distance metrics on vectors such as the inner product. A privacy-preserving music information retrieval system is given in follow-up work by Wang & Zhao [51] using these techniques. Evaluation is performed on $1,000$ audio files using YAMNet embeddings, showing that PHE for encrypted queries achieves high accuracy with low runtime and memory overhead than Somewhat Homomorphic Encryption (SHE). In contrast, PRISM demonstrates that by leveraging SHE with ciphertext packing [9], privacy-preserving semantic search can scale effectively to near-billion item corpora. Additionally, PRISM introduces a permutation protocol to prevent dataset reconstruction from repeated queries, and uses lattice-based SHE to provide plausible post-quantum security—a critical advantage over standard PHE schemes.

Our work aims to bridge these gaps by combining the semantic power of vision–language embeddings with efficient homomorphic encryption techniques and system-level protocols (permutation, hierarchical clustering, and asynchronous pub/sub) to deliver private, scalable, and decentralised natural-language search and discovery over distributed media datasets.

3. Privacy-Preserving Semantic Matching

3.1. Homomorphic Encryption

Homomorphic Encryption (HE) is a privacy-preserving cryptographic primitive that enables certain mathematical operations to be performed directly on encrypted data (ciphertexts). The results of these operations remain encrypted and, upon decryption, are mathematically identical to the output of the same operations executed on the corresponding plaintexts (the original, unencrypted data). In the context of private retrieval, HE enables a data owner to compute the distance—specifically, the inner product, which acts as an exact proxy for cosine similarity on normalised vec-

tors (as detailed in Sec. 3.2)—between a user's encrypted query embedding and its own plaintext database embeddings without ever observing the query's contents. Partially Homomorphic Encryption (PHE) supports group homomorphism (addition or multiplication), while Somewhat Homomorphic Encryption (SHE) and Fully Homomorphic Encryption (FHE) support ring homomorphism (addition and multiplication). All HE variants admitting addition enable inner product computation via linearity, a property often exploited in privacy-preserving machine learning.

PRISM uses SHE, which consists of four algorithms: a key generation algorithm $(\texttt{pk}, \texttt{sk}) \leftarrow \texttt{KeyGen}(\kappa, d)$, which generates key material given a security parameter κ and a maximum multiplicative depth d (where $d = 0$ for PRISM); an encryption algorithm $c \leftarrow \texttt{Enc}_{\texttt{pk}}(m)$, which encrypts a message m under the public key $\texttt{pk}$ to produce a ciphertext; a decryption algorithm $m \leftarrow \texttt{Dec}_{\texttt{sk}}(c)$, which decrypts a ciphertext c using the secret key $\texttt{sk}$; and an evaluation function $c \leftarrow \texttt{Eval}(\texttt{op}, c_1, c_2)$ which takes an operation $\texttt{op} \in \{+, \cdot\}$ and two ciphertexts c_1 and c_2 and outputs a ciphertext c. The scheme is homomorphic in the sense that given messages m_1 and m_2 and corresponding ciphertexts c_1 and c_2, $\texttt{Dec}_{\texttt{sk}}(\texttt{Eval}(+, c_1, c_2)) = m_1 + m_2$ and similarly for multiplication. See e.g. [18] for formal security and correctness definitions.

Many S/FHE schemes based on the hardness of lattice problems support Single-Instruction Multiple Dataset operations through a technique called ciphertext packing, where a *single* ciphertext is an encrypted vector of plaintext values [9], instead of one per vector component (as would be needed when using PHE). This not only obviates the need for expensive group exponentiations per vector component, but also drastically reduces the communication overhead when transferring ciphertexts. While PHE generally involves much smaller public keys than S/FHE, we found in experiments that this cost was outweighed by the fact that only one ciphertext needs to be sent. Another advantage of lattice-based S/FHE schemes is that they are generally believed to be resistant against quantum attacks, whereas most (if not all) mainstream PHE schemes rely on the hardness of discrete logarithms or prime factorisation.

We use the CKKS (HEAAN) cryptosystem [18] since it natively supports real-valued vectors and because we do not require exact computations. We chose TenSEAL to implement our protocol which supports tensor operations over ciphertexts in the SEAL library. SEAL provides functions for de/serialising public keys and ciphertexts, which is essential for transmitting the data over the network. Details on parameter choices are provided in Sec. 5.2. Throughout, wherever we write that a ciphertext is sent over the network, we assume the data structure includes any auxiliary information required to manipulate the ciphertext such as the public key and associated parameters.

3.2. Vision-Language Models

Advancements in modern deep learning have led to the proliferation of multimodal models—particularly Vision-Language Models (VLMs)—capable of embedding distinct data types into a common high-dimensional vector space in which geometric proximity represents semantic similarity [7]. Our system leverages the Contrastive Language-Image Pretraining (CLIP) foundational VLM architecture [43]. CLIP trains two parallel encoders—one for text and one for images—to map inputs into a shared d-dimensional embedding space. This process involves training the weights of a neural network to maximise the cosine similarity between associated text and image pairs:

$$\cos(\theta) = \frac{\langle \vec{a}, \vec{b} \rangle}{\|\vec{a}\|_2 \cdot \|\vec{b}\|_2}$$

for each pair of vectors $(\vec{a}, \vec{b})$, where $\vec{a}$ is the embedding vector of a text prompt, $\vec{b}$ is the embedding vector of a corresponding training image, and θ is the angle between them. CLIP aims to minimise the cosine similarity between unmatched pairs while maximising the similarity between matched pairs (hence 'contrastive').

In order to find images matching a text prompt, we search for database items whose embedding vectors are close to the query vector with respect to cosine similarity. A critical observation for our privacy-preserving protocol is that if all vectors are L_2-normalised (i.e., scaled such that $\|\vec{v}\|_2 = 1$) prior to encryption, the denominator in the equation above becomes 1. Consequently, the cosine similarity reduces strictly to the inner product, where $\hat{a}$ and $\hat{b}$ are the normalized query and database vectors, respectively: $\text{sim}(\hat{a}, \hat{b}) = \langle \hat{a}, \hat{b} \rangle$.

For the embedding layer, we chose the open-source CLIP ViT-B/16 model architecture [43] due to its strong zero-shot transfer performance which ensures our results are directly comparable to established retrieval baselines [38]. Furthermore, the model produces 512-dimensional embeddings, which enables a critical balance for our system: it preserves high semantic fidelity while ensuring that the encrypted inner-product operations remain computationally lightweight. We enforce L_2-normalisation on all model outputs to satisfy the requirements of our similarity protocol.

While we use CLIP to evaluate PRISM, this geometric approach is modality-agnostic. PRISM naturally extends to any data types where a joint embedding model exists.

3.3. Security model

We prove our protocols secure in the Universal Composability (UC) framework introduced by Canetti [13], as is standard practice for cryptographic protocols involving multiple parties. Protocols secure in the UC framework are secure

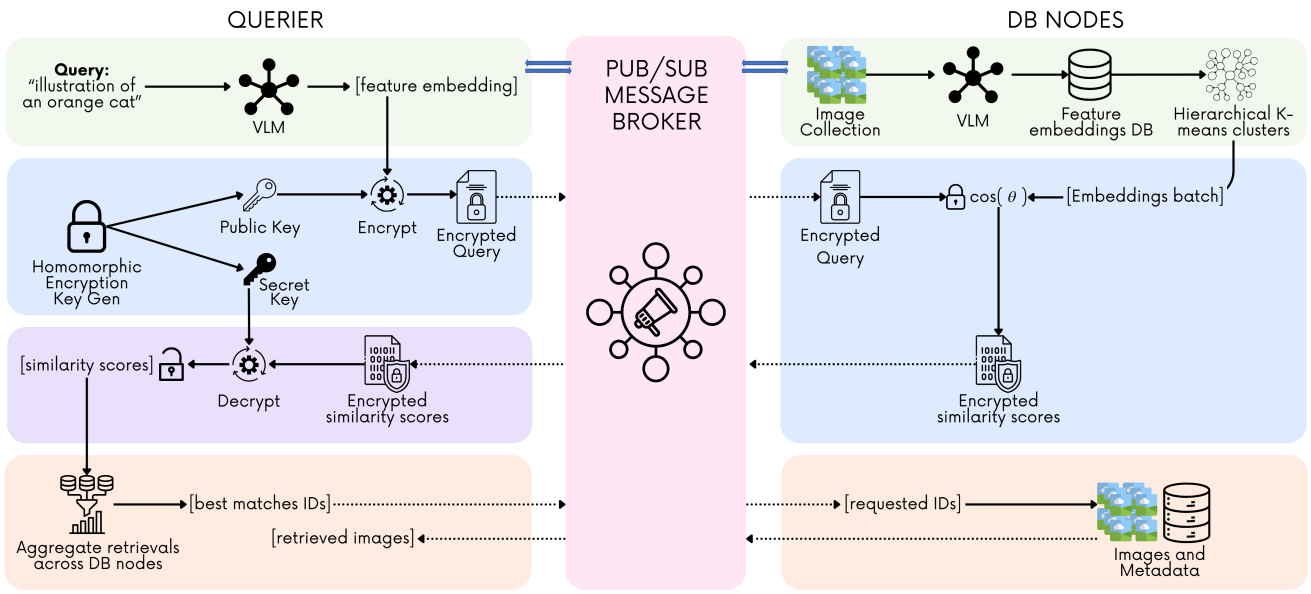

Figure 1. Architecture of the PRISM privacy-preserving distributed search protocol. The pipeline consists of four stages: **(Green)** VLM feature extraction and data indexing; **(Blue)** Query encryption and encrypted similarity computation; **(Purple)** Client-side score decryption; and **(Orange)** Final retrieval of image metadata for the top-ranked results.

even when arbitrary instances of the same or other protocols are run in serial or parallel. The reader is referred to the supplementary material for the proofs.

4. Distributed Search Protocol

This section details the PRISM architecture (Fig. 1) for secure, scalable semantic discovery over independent data silos.

4.1. Communication Mechanism

For communication between parties, our architecture uses an asynchronous publish/subscribe ('pub/sub') model. Clients broadcast encrypted queries to a network of distributed data nodes and subscribe to the corresponding response channels. Within decentralised web architectures, this messaging layer is naturally supported by privacy-preserving, peer-to-peer routing protocols——such as Waku [49] for censorship-resistant message relaying, or mixnets [16] and Oblivious HTTP [33] to decouple client IP addresses from request metadata and thwart traffic analysis.

Because the underlying transport infrastructure may be operated by untrusted peers, we do not depend on it for authentication or data confidentiality. We rely on SHE to protect query confidentiality; privacy leakage is described in the supplementary material. The protocol is secure *with abort*: an adversary with network control may drop or corrupt messages, causing a query to fail, but the client will simply reject invalid responses and halt the protocol. This models decentralised environments where nodes frequently drop packets or disconnect.

4.2. Index Construction and Clustering

A linear scan of billion-scale datasets is computationally infeasible. We reduce costs by organising the database into an Inverted File-type hierarchical index, which significantly prunes the search space.

Spherical K-Means Clustering. As established in Section 3.2, our embedding space lies on the unit hypersphere $\mathbb{S}^{d-1}$, where semantic similarity is defined by angular proximity. Standard K-Means clustering minimises within-cluster Euclidean distortion, which induces planar Voronoi partitions that are suboptimal for directional data. We therefore employ Spherical K-Means clustering [25], which enforces a unit-norm constraint on centroids and optimises the clustering objective by maximising the cosine similarity between data vectors and their assigned centroids.

To facilitate retrieval at scale, a partition-based hierarchical index is implemented using the FAISS library [26]:

1. **Training:** We initialise the index by training cluster centroids via spherical k-means using a random sample of the dataset empirically fixed at 5×10^5 vectors.
2. **Assignment:** Once centroids are fixed, the system assigns each vector to the nearest centroid based on cosine similarity.
3. **Hierarchical Structure:** The implementation supports a configurable index depth L. For depths $L > 1$, the system employs recursive 'centroid-of-centroids' clustering

to provide fine-grained sub-clusters.

During a retrieval session, the querier and data owner engage in a tree traversal down to the matching data items. To maintain security, a new public key is generated for each session, though the same key is used throughout a single tree traversal. The impact of this clustering approach on retrieval accuracy is evaluated in Sec. 5.2.

4.3. Matrix / Vector Multiplication Protocol

The core of our distributed search architecture relies on a secure matrix/vector multiplication protocol. The end goal of the system is to facilitate consent-driven data discovery; thus, data owners are inherently incentivised to behave honestly to successfully match with clients and license their media. The retrieval follows a hierarchical four-stage protocol to ensure scalability:

1. **Query Encryption & Broadcast:** The client computes an L_2-normalised CLIP vector $\vec{v}$ of their query, generates a key pair $(\mathtt{pk}, \mathtt{sk}) \leftarrow \mathtt{KeyGen}(\kappa)$, and broadcasts the packed ciphertext $\vec{c} \leftarrow \mathtt{Enc}_{\mathtt{pk}}(\vec{v})$ to the network.
2. **Encrypted Similarity Computation:** Each data node homomorphically computes the matrix-vector product $\vec{c'} \leftarrow M \cdot \vec{c}$, where M is a plaintext matrix of its normalised cluster centroids. The resulting encrypted cosine similarity scores ($\vec{c'}$) are returned to the client.
3. **Selection & Refinement:** The client decrypts the scores ($\vec{v'} \leftarrow \mathtt{Dec}_{\mathtt{sk}}(\vec{c'})$) to identify the top-$k$ nearest centroids. It then requests a fine-grained search against the specific embeddings within those top-ranked partitions.
4. **Retrieval:** Upon decrypting the scores and identifying the closest matches, the client requests the canonical metadata (e.g., watermarked images, licensing contracts) for those specific items from the data owners.

4.4. Permutation Protocol

It is known that it is possible to reverse-engineer images from embedding vectors to some extent [17]. Iterated invocations of the search protocol could allow a corrupt client to learn items in an honest data owner's dataset by sequentially submitting linearly independent vectors over a number of queries. This fact is inherent to any protocol that allows one party to learn inner products between private inputs, and it would likely require heavy machinery (e.g. full Multi-Party Computation protocols) to allow parties only to learn indices of the top matches.

In our implementation, we enforce that data owners permute the indices of their input datasets (i.e. matrix rows) on each query and impose minimum dataset sizes (padding with random dummy vectors if necessary, which has negligible impact on performance) to avoid subset-sum-style attacks, to prevent the client from constructing a constrained set of equations to learn dataset embedding vectors. Attack analysis is given in the supplementary material.

5. Evaluation

5.1. Experimental Setup

To evaluate the performance of our system, we use widely accepted retrieval benchmarks as well as a large-scale image-caption dataset:

- MS-COCO captioning 2014[40] (Karpathy test split [34])
- Flickr30k [52] (Karpathy test split [34])
- UIUC Pascal sentences [45] derived from [28]
- Recap-DataComp-1B Dataset [39]

We chose Recap-DataComp-1B due to its large scale and its recaptioned, semantically richer captions, which improve alignment between images and text and have been shown to benefit cross-modal tasks such as CLIP training and retrieval [39]. To evaluate our system at scale, we construct corpora of varying sizes (250M, 500M and the full dataset). For each corpus size, we generate a dedicated test set of 500 queries using a stratified sampling approach to ensure diversity in query difficulty. We first identify the top 1 million image-caption pairs within the specific corpus based on cosine similarity. From this leaderboard, we sample 200 pairs from the very top ranks, 200 starting at the 500,000$^{\text{th}}$ rank, and the final 200 pairs near the 1 millionth rank. As massive web-scraped datasets often contain duplicates that can artificially lower recall scores (by scoring higher than the designated ground truth item), we perform a validation search for these candidates and remove any query where a duplicate item in the index outscores the ground truth. Finally, we truncate the remaining valid candidates to a fixed set of 500 queries per corpus size for the evaluation.

First, we perform a baseline experiment using the smaller benchmark datasets to isolate and measure the impact of encryption on retrieval accuracy. Next, we evaluate the system by arbitrarily splitting the Recap-DataComp-1B dataset up and emulating a number of data owner nodes responding to client queries. In all experiments, the dataset partitions are equally distributed across nodes and all nodes cluster the local data partition into an equal number of centroids (K). We evaluate the system in a variety of setups to demonstrate robustness, accuracy, and scalability.

We report the average end-to-end latency per query, decomposed into *MatMul time* (the computational cost of homomorphic inner products) and *I/O time* (data and metadata loading and general I/O overheads), alongside the *global minimum and maximum* latencies across all nodes.

All experiments were performed on a cluster of three Linux servers running Ubuntu 18.04 LTS. Two nodes have 14-core Intel Core i9-10940X CPUs and the third node has an Intel Core i7-14700K CPU. Each server has 128 GB RAM. The dataset is hosted on a local RAID array attached to one server and mounted via NFS on the other servers; to minimise network I/O latency during retrieval, the relevant vector index shards are cached in memory on each server.

5.2. Cryptographic Overhead and Performance

Stage	Actor	Operation	Size	Time
1. Setup	Client	Context Gen. *(One-time)*	-	50 ms
	Client	Pub. Key *(Saved by each DB)*	18.3 MB	-
	Client	Sec. Key *(Kept Local)*	535 KB	-
2. Query	Client	Encryption *(Per Query)*	230 KB	9 ms
3. Similarity	Server	Calc. Similarity Scores *(batch)*	129 KB	306 ms
Scoring	Client	Decrypt Similarity Scores *(batch)*	129 KB	10 ms

Table 1. Cryptographic costs by pipeline stage. *Note: Similarity operations are performed in batches of 4096 items.*

We used the CKKS parameters from the SEAL examples, with polynomial modulus degree 8192 (giving us 4096 slots per vector batch) but do not require homomorphic multiplication and so chose moduli of bit sizes 60, 40 and 60, totalling 160 bits) and hence are well within 128-bit quantum security. We explored other parameter choices, such as setting polynomial modulus degree to $16,384$ (it is required to be a power of 2), but found that the choice above gave the best balance between data transferred over the network (a larger modulus leads to a larger public key) and number of additional rounds of computation required.

We first evaluate the computational and storage overhead imposed by the homomorphic encryption scheme and report them in Table 1. The SHE parameters were selected to provide (at least) 128-bit quantum security with a good balance between with vector dimension and public key size. The values reported are directly tied to this specific configuration; altering the cryptographic parameters would result in variations in both storage sizes and computation times.

The system incurs a negligible one-time setup cost (≈ 50 ms). The public key (≈ 18 MB) is transmitted once to each data node upon joining, while the compact secret key (535 KB) remains local to the client. The primary latency source is the server-side similarity calculation (averaging ≈ 306 ms per batch in our experiments), but it is worth noting that the server-side computation time is naturally dependent on available CPU resources and system load. This computation is highly bandwidth-efficient: the server compresses 4,096 encrypted scores into a single 129 KB payload, allowing the client to decrypt the entire batch in roughly 10 ms.

These results demonstrate that the computation and storage burden is sufficiently low for real-time applications. The bandwidth requirements (kilobytes per query/response) and latency (sub-second processing) indicate that the system can operate effectively at scale without becoming bottlenecked by the privacy-preserving layer.

5.3. Encrypted Retrieval Accuracy Parity

We first isolate the cryptographic layer to validate that the approximation errors inherent in the CKKS homomorphic encryption scheme do not degrade retrieval accuracy. To establish this baseline, we perform exact retrieval via a full linear scan (without prior clustering or dataset permutation) on the MS-COCO, Flickr30k, and UIUC Pascal datasets. We conduct these searches twice: once in standard plaintext and once using our encrypted protocol.

Table 2. Retrieval metrics for benchmark datasets. We report Recall@K (R@1, R@5, R@10) as percentages. Results are identical for both plaintext and encrypted retrieval.

Dataset	R@1	R@5	R@10	MRR
Flickr30k	62.18	85.72	91.92	0.7255
MS-COCO	33.09	58.40	68.99	0.4514
Pascal Sentences	58.54	84.37	91.76	0.6992

As shown in Table 2, our encrypted retrieval protocol achieves results identical to standard plaintext CLIP baselines. This confirms that the CKKS approximation error is negligible for retrieval tasks and that the privacy layer does not degrade accuracy. Our baseline results are consistent with those reported in prior work using CLIP ViT-B/16 on these benchmarks [22].

5.4. Clustering Depth and Granularity

We first determine the optimal clustering depth and density by fixing the corpus size at 250M items distributed across 5 nodes (50M items per node) restricting the search to the top 10 closest clusters per query.

5.4.1. Hierarchical Clustering Depth

We first evaluate the impact of hierarchical clustering depth on accuracy and system latency. We compare 1, 2, and 3-level hierarchical indices across different branching factor configurations. As shown in Table 3, increasing the hierarchy depth resulted in a significant drop in retrieval accuracy and a slight increase in latency. The latency increase is likely due to the overhead of sequential cryptographic rounds outweighing the benefits of search space pruning at this specific data scale. However, hierarchical indexing remains a critical direction for future investigation at much larger scales (e.g., multi-billion item corpora), where the linear complexity of scanning a flat list of clusters would become a bottleneck. Consequently, a flat (Depth 1) clustering architecture is adopted for the subsequent experiments.

5.4.2. Clustering Granularity

Next, we determine the optimal clustering density by varying the number of clusters (K) per node from 4,000 to 16,000 using a flat index (Table 3, top rows). As expected, finer-grained clustering (higher K) significantly reduces latency. Moving from 4k to 16k clusters reduces the average end-to-end latency by nearly half (13.37s to 6.95s), a speedup driven primarily by the reduction in the number of vectors—and consequently the number of encrypted

Table 3. Impact of clustering granularity and depth on retrieval performance. Increasing the number of clusters (K) per node (5 total) reduces the search space per query, significantly lowering latency (13.37s $\rightarrow$ 6.95s) but at the cost of reduced recall (R@1 drops 73.0% $\rightarrow$ 61.0%). We select $K = 8,000$ as the optimal configuration for depth = 1. **Bottom (Depth 2/3):** Deeper hierarchies show increased latency and reduced accuracy compared to the flat baseline in this configuration.

		Recall Accuracy (%)				Avg. Latency Breakdown (s)			Total Time Range (s)	
Depth	**Clusters ($L_1 \times \ldots$)**	**R@1**	**R@5**	**R@10**	**R@100**	**Total End-to-End**	**MatMul**	**I/O**	**Min**	**Max**
1	4,000	73.0	76.6	76.8	77.0	13.37	7.90	5.27	5.68	29.38
	8,000	67.8	71.4	71.6	71.6	10.10	5.28	4.73	4.18	20.28
	12,000	62.8	65.4	65.6	65.8	7.71	4.00	3.63	4.04	18.74
	16,000	61.0	64.0	64.2	64.4	6.95	3.61	3.27	3.74	14.50
2	$6,000 \times 300$	36.4	37.2	37.4	37.4	15.37	9.05	6.22	2.89	47.83
	$10,000 \times 500$	33.2	33.6	33.6	33.6	12.66	6.89	5.69	2.62	43.67
3	$12,000 \times 120 \times 20$	33.6	34.6	35.0	35.0	13.68	8.54	5.04	3.96	49.30
	$16,000 \times 160 \times 20$	25.6	25.6	25.8	25.8	17.12	8.67	8.37	3.68	52.72

batches—processed during homomorphic matrix multiplication for inner product calculation. However, this efficiency comes at the cost of retrieval accuracy; because the fixed search strategy probes a constant number of centroids, increasing K dilutes the search space, causing Recall@1 to drop from 73.0% at 4k clusters to 61.0% at 16k. We observe significant variance in the end-to-end timings, largely attributable to the varying density of the retrieved clusters, which leads to disparate I/O and processing times depending on whether a 'heavy' or 'light' cluster is queried.

Based on these findings, we selected 8k clusters as the optimal operating point for the 50M-per-node partition size, providing a balance between precision and latency.

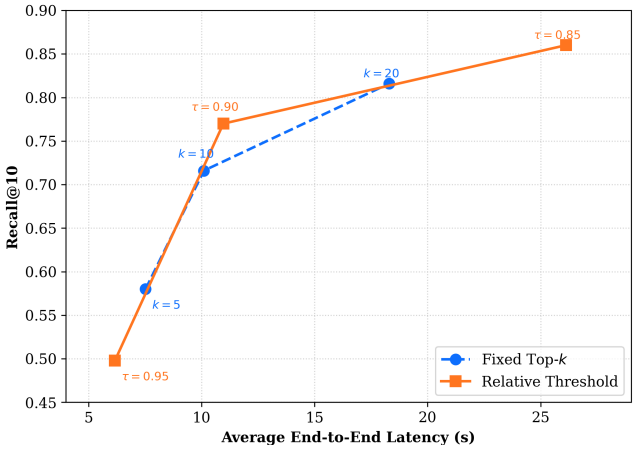

Figure 2. Recall@10 vs. Latency trade-off for Fixed (dashed blue) and Relative (solid orange) cluster selection strategies. The Relative strategy ($\tau = 0.90$) achieves superior accuracy (0.77 vs 0.716) compared to the Fixed strategy ($k = 10$) at a similar latency budget ($\approx$ 10s), demonstrating the efficiency of adaptively selecting clusters based on score distribution.

5.5. Cluster Selection Strategy

We evaluate two cluster selection strategies to optimise recall versus computational cost. In **Fixed top-k search**, the client selects a constant number of highest-scoring centroids. The disadvantage of this approach is that it ignores the data distribution, excluding relevant clusters when a query has many potential close matches. Conversely, **Relative search** dynamically selects all clusters satisfying $S_i \geq S_{max} \times \tau$, where S_{max} is the highest similarity cluster score and τ is a similarity threshold. In practice, we enforce bounds on this selection (between 1 and 100 clusters), allowing the system to adapt to query ambiguity and data distribution while preventing network flooding.

We evaluate the efficiency of the **Fixed** versus **Relative** search strategies by measuring the trade-off between retrieval accuracy and end-to-end latency. Using the 250M corpus, we vary the search parameters:

1. **Fixed:** Top-$k \in \{5, 10, 20\}$ leaf clusters.
2. **Relative:** Threshold $\tau \in \{0.85, 0.90, 0.95\}$ of the max similarity score.

Figure 2 illustrates the results. The relative search strategy (solid orange) outperforms the fixed strategy (dashed blue) by providing a superior accuracy-latency trade-off. At a comparable latency budget of $\approx$10–11 seconds, the Relative strategy ($\tau = 0.90$) achieves a Recall@10 of 0.77 in 10.9s, significantly higher than the Fixed strategy ($k = 10$), which reaches a Recall@10 of 0.716 in 10.1s. This efficiency stems from the adaptive nature of the relative threshold: it allocates computational resources to ambiguous queries (searching more clusters) while aggressively pruning the search space for distinct, high-confidence queries.

Therefore, we select the Relative strategy with $\tau = 0.90$ as the optimal configuration for subsequent experiments.

Table 4. System Scalability Evaluation. **Top (Network Scaling):** Distributing a fixed 250M corpus across increasing nodes (N) improves both recall accuracy and retrieval latency. **Bottom (Corpus Scaling):** Scaling the dataset to 500M and 745M (maintaining 50M items/node) demonstrates consistent performance.

				Recall Accuracy (%)				Avg. Latency Breakdown (s)			Total Time Range (s)	
Experiment	**Corpus**	**Nodes**	**Clusters (K)**	**R@1**	**R@5**	**R@10**	**R@100**	**Total E2E**	**MatMul**	**I/O**	**Min**	**Max**
Network Scaling	250M	3	13,330	68.2	71.0	71.2	71.4	15.11	8.50	6.40	2.09	75.83*
	250M	5	8,000	73.0	76.6	77.0	77.0	10.96	6.53	4.29	1.43	69.32*
	250M	10	4,000	74.8	78.4	78.8	79.0	10.18	6.28	3.77	0.97	65.52*
	250M	20	2,000	75.8	79.4	79.8	80.0	13.69	6.52	7.08	1.98	93.50*
Corpus Scaling	250M	5	8,000	73.0	76.6	77.0	77.0	10.96	6.53	4.29	1.43	69.32*
	500M	10	8,000	70.0	72.8	73.0	73.0	16.06	8.78	7.10	2.73	74.20*
	745M	15	8,000	71.0	71.4	71.6	71.6	17.33	7.82	9.40	3.28	72.66*

* Outlier latency observed when the relative search strategy selected the maximum cap of 100 clusters for ambiguous queries.

5.6. System Scalability

Table 4 summarizes our evaluation of the system's scalability across two dimensions: network size (decentralisation) and total data volume (corpus size).

5.6.1. Decentralisation (Network Scaling)

First, we assess performance as a function of network size by distributing the fixed 250M corpus across an increasing number of nodes ($N \in \{3, 5, 10, 20\}$). To maintain the optimal data-to-cluster ratio determined previously, the number of clusters per node (K) is scaled inversely to the local partition size, ranging from 13,330 clusters (3 nodes) down to 2,000 clusters (20 nodes).

As shown in the top rows of Table 4, retrieval accuracy increases with the fragmentation of the index. Recall@1 improves monotonically from 68.2% with 3 nodes to 75.8% with 20 nodes. This suggests that querying multiple smaller, locally optimised indices retrieves a higher quality set of candidates than querying fewer, larger monolithic indices.

Latency initially improves with parallelisation, dropping from 15.11s (3 nodes) to 10.18s (10 nodes) as per-node load decreases. However, it rebounds to 13.69s at 20 nodes, likely due to the increased aggregate load on our physical simulation servers when coordinating a larger number of database nodes.

5.6.2. Corpus Size (Data Scaling)

Finally, we evaluate the system's capacity to handle massive datasets by scaling the total corpus from 250M, to 500M, and to the full available dataset (745M). This final figure represents the total dataset accessible at download time, reflecting the typical attrition rate of massive URL-based web scraped datasets. To maintain an optimal data density ratio, we scale the infrastructure linearly, employing 10 nodes for the 500M corpus and 15 nodes for the 745M corpus, with each node indexing its partition into 8,000 clusters.

The bottom rows of Table 4 demonstrate that while the system maintains high recall (71.0% R@1), the end-to-end latency increases to 17.33s due to the volume of data processed. Notably, the I/O overhead grows significantly, accounting for 9.40s of the total time. We explicitly report the breakdown between MatMul and I/O time to highlight that this bottleneck is largely **implementation-specific** rather than cryptographic; in a production environment with optimised storage layers and further parallelisation, these costs could be drastically reduced.

Our evaluation confirms that privacy-preserving retrieval is viable at the near-billion scale. We recommend using a Relative Search strategy ($\tau \approx 0.90$) to maximize recall efficiency and a flat index with 8,000 clusters for 50M-item partitions. Finally, since storage I/O acts as the primary bottleneck at scale, future deployments should prioritize high-throughput storage solutions to fully leverage the architecture's parallelisation capabilities.

6. Conclusion

We presented PRISM, a near-billion-scale, privacy-preserving semantic search protocol designed to enable confidential media discovery across distributed, sovereign image archives. We demonstrate feasibility using the Recap-DataComp-1B dataset as a benchmark. We used the CKKS Somewhat Homomorphic Encryption (SHE) scheme as implemented in the TenSEAL library, showing that SHE can be effectively deployed to bolster privacy without compromising on accuracy. Our experimental results confirm the system's scalability: we successfully searched a distributed index of 250M items in approximately 10.18 seconds and the full 745M dataset in 17.33 seconds. PRISM addresses a critical gap in the decentralised creative economy. By allowing creators and data owners to federate and expose their collections to natural-language search without inadvertently releasing their raw media to the public domain or AI web-scrapers, this protocol provides a secure foundation for consent-driven discovery and fair data monetisation.

Acknowledgment This work was supported by DECaDE under UKRI/EPSRC grant EP/T022485/1.

7. Supplementary Material

7.1. Security proofs

We prove our protocols secure in the Universal Composability (UC) framework introduced by Canetti [13], as is standard practice for cryptographic protocols involving multiple parties. Protocols secure in the UC framework are secure even when arbitrary instances of the same or other protocols are run in serial or parallel, which is not the case for standalone security. We assume the reader is familiar with proofs in this framework, and consequently provide only a high-level review in this subsection.

UC security is proved as follows. First, one defines an ideal version of the protocol called the functionality, which acts as a trusted entity interacting with all parties to achieve some specified outcome. Then one creates a protocol between the parties that approximates the ideal version to achieve the same goal. Finally, one shows that for any adversary against the protocol, there is a simulator against the functionality such that any environment, in which the adversary operates, cannot tell if it is interacting as in the real-world protocol execution or in the ideal-world functionality execution. The inability to distinguish between these executions implies no adversary can learn anything from a protocol execution that cannot be learnt when instead interacting with the functionality, which is secure by construction.

At a high level, the UC composability guarantee says interacting with a trusted functionality is equivalent to executing a protocol that UC-securely realises it. This means the UC framework allows for cleaner exposition by supporting a modular approach to building complex protocols. When developing a protocol Π to realise functionality $\mathcal{F}$, we may make calls to any ideal functionality $\mathcal{F}'$ that can be UC-securely realised by some subprotocol Π'. This simplifies the process of proving security for Π since we no longer need to reason about computation in Π', instead dealing with ideal calls to $\mathcal{F}'$.

Throughout, we refer to the client performing queries as the 'receiver', and the data owners as 'senders', following standard conventions in the literature for comparable multi-party protocols.

Notation. We use κ to denote the computational security parameter. A variable x is assigned a value X by the notation $x \leftarrow X$. For a set X, we denote its cardinality by $|X|$. An arbitrary field is denoted by $\mathbb{F}$ and an n-dimensional column vector by $\vec{v} \in \mathbb{F}^n$. For a matrix $M \in \mathbb{F}^{m \times n}$ and a vector $\vec{v} \in \mathbb{F}^n$, we denote by $M \cdot \vec{v}$ the standard matrix/vector multiplication.

7.2. Communication Mechanism

Since the messaging system is not our focus, we do not provide the protocol and prove its security. The functionality we describe, given in Figure 3, is very weak (the adversary has considerable power: there is no guaranteed message delivery, authentication, or confidentiality) and far stronger messaging protocols have been implemented and proved UC secure, e.g. the Signal protocol [14]. We do not require authentication because it allows data owners not to reveal their identities until queriers request specific images. We essentially rely on the fact that the client messages are ciphertexts for confidentiality, and that if these ciphertexts are corrupted then the clients will usually halt the protocol.

$\underline{\mathcal{F}_{\mathsf{PubSub}}}$

The ideal-world adversary $\mathcal{S}$ may corrupt any entity at any point and prevent publishing any message or delivery of a message to a subscriber.

Initialise On input a message $(\texttt{sid}, \texttt{init})$, if it is the first message received, store `sid` and ignore any further messages with a different `sid`, and initialise an empty record $R \leftarrow \varnothing$.

Subscribe On input a message $(\texttt{sid}, \texttt{subscribe}, p)$ from player with identity `id`, store the entry $(\texttt{id}, p)$ in R.

Publish On input a message $(\texttt{sid}, \texttt{publish}, m, p)$ from any player, let $I = \{\texttt{id} : \exists(\texttt{id}, p) \in R\}$ and send the message $(\texttt{sid}, \texttt{new\_message}, m, p)$ to all players in I.

Figure 3. Functionality $\mathcal{F}_{\mathsf{PubSub}}$

$\underline{\mathcal{F}_{\mathsf{Mv}}}$

Query Await a message $(\texttt{query}, \vec{v})$ from the receiver and then do the following for each sender, indexed by $\texttt{id}$:

- Await a matrix $M_{\texttt{id}} \in R^{m \times n}$, where R is a ring, and if they are corrupt await errors $\vec{\varepsilon_{\texttt{id}}}$ from $\mathcal{S}$.
- Compute $\vec{v'_{\texttt{id}}} \leftarrow M'_{\texttt{id}} \cdot \vec{v} + \vec{\varepsilon_{\texttt{id}}}$.
- Send $\vec{v'_{\texttt{id}}}$ to the receiver.

Figure 4. Matrix / vector multiplication functionality

7.3. Matrix / Vector Multiplication Protocol

Our distributed search protocol involves the use of (a variant of) the functionality $\mathcal{F}_{\mathsf{Mv}}$ (see Figure 4) between one client 'receiver' and multiple data owner 'senders' that allows the receiver to obtain the image of their vector under a linear map (matrix under some choice of basis) held by the senders.

The protocol for $\mathcal{F}_{\mathsf{Mv}}$ could be built in many ways. One possibility is (Vector) Oblivious Linear function Evaluation ((V)OLE), which exactly enables one party to learn the image of their input under an affine linear function chosen by the other [3]. However, we prefer a solution that minimises communication rounds and allows multiple data owners to respond to a single client query so that the client query communication is a oneshot message. This better reflects the asynchronous nature of real-world distributed communication where users are not always online at the same time. VOLE protocols would also generally require preprocessing per client on the part of the data owners. Nevertheless, we leave investigation of a VOLE-based protocol as an interesting direction for future work. New directions in Private Set Intersection (PSI) and Private Information Retrieval (PIR) protocols may also prove fruitful for our use case [15].

Unfortunately, we cannot prove UC security of the 'clean' $\mathcal{F}_{\mathsf{Mv}}$ functionality using our homomorphic encryption approach, essentially because when senders are corrupt, the simulator is unable to extract the adversary's matrices without rewinding (which is forbidden in the UC framework). To sidestep this difficulty, we instead define a modified functionality $\mathcal{F}^{\kappa}_{\mathsf{Mv}}$ that leaks cryptographic artefacts (public keys and ciphertexts) to the simulator. (The same approach is taken for the proofs in the celebrated SPDZ protocol [23, 24].) We would expect to be able to bootstrap our protocol to satisfy the cleaner functionality $\mathcal{F}_{\mathsf{Mv}}$ by adding commitments and zero-knowledge proofs (ZKPs), but our implementation would then have to use black-box ZKP techniques due to the limited availability of software libraries, which would significantly increase the computational overhead of our protocol, compromising our goal of billion-scale search. Since ZKPs for lattice-based cryptography are being actively researched (e.g. [29]), we anticipate that, in time, efficient proofs will become available to allow us to create a protocol for the cleaner functionality.

We emphasise that the end goal of our distributed search protocol is for data owners to sell the rights to images for successful matches, so they are generally incentivised to behave honestly (a client will stop the interaction if the matches are not close enough to their input message). The functionality $\mathcal{F}^{\kappa}_{\mathsf{Mv}}$ therefore only (implicitly) offers passive security; in particular, the adversary could use the public parameters provided with a query ciphertext to encrypt *any* vector as a response. Indeed, an adversarial data owner can always refuse to hand over data items to the client at the last step. There does not, however, appear to be a reasonable incentive for doing so except to disrupt the system, which can be achieved more easily via non-cryptographic attacks (such as denial-of-service attacks). Mitigating these problems in an *asynchronous* protocol seems to be a complex task, potentially requiring commitments and Zero-Knowledge Proofs (ZKPs) and/or escrow systems (e.g. smart contracts). By contrast, our approach is extremely lightweight and scalable.

The advantage of realising $\mathcal{F}_{\mathsf{Mv}}$ rather than $\mathcal{F}^{\kappa}_{\mathsf{Mv}}$ would be that our proof would admit a reduction to the problem of breaking IND-CPA security of the homomorphic encryption scheme [8] directly; at present, the functionality is vulnerable to non-black-box attacks on the encryption scheme. In brief, realising $\mathcal{F}_{\mathsf{Mv}}$ requires an IND-CPA homomorphic encryption scheme because the simulator needs to send a ciphertext to the adversary on behalf of an honest receiver without revealing the fact that the plaintext does not correspond to the receiver's input vector (since this is not revealed to the simulator by $\mathcal{F}_{\mathsf{Mv}}$).

In our description of $\mathcal{F}^{\kappa}_{\mathsf{Mv}}$, the client is the receiver, and the data owners are the senders. For simplicity of exposition, we assume that a ciphertext $\vec{c}$ is a data object that includes any public information required to manipulate it under the homomorphic encryption scheme (e.g. ring moduli). Our protocol Π_{Mv} to realise $\mathcal{F}^{\kappa}_{\mathsf{Mv}}$ in the $\mathcal{F}_{\mathsf{PubSub}}$-hybrid model is given

$\underline{\mathcal{F}^{\kappa}_{\mathsf{Mv}}}$

The functionality is parameterised by a computational security parameter κ, which bounds the environment's computational complexity. Once the functionality has been initialised on receipt of any message, the `sid` from this message is used as the session ID and all future messages with different `sid` are ignored. If an honest party receives the message $\perp$ at any point, they abort the protocol.

Query If the receiver is honest, do the following:

- Await a message $(\texttt{sid}, \texttt{query}, \vec{v})$ from the receiver, where $\vec{v} \in \mathbb{C}^n$ for some n.
- Sample a new key pair $(\texttt{pk}, \texttt{sk}) \leftarrow \texttt{KeyGen}(\kappa)$ for a homomorphic encryption scheme and leak a ciphertext $\vec{c} \leftarrow \texttt{Enc}_{\texttt{pk}}(\vec{v})$ to $\mathcal{S}$. (Recall that we assume the ciphertext data structure embeds public parameters including the public key $\texttt{pk}$.)
- Await input matrices $M_{\texttt{id}}$ from honest senders and compute $\vec{v}_{\texttt{id}} \leftarrow M_{\texttt{id}} \cdot \vec{v}$.
- Await response vectors $\vec{c'_{\texttt{id}}}$ from $\mathcal{S}$ on behalf of each corrupt sender (if any), and attempt to decrypt, $\vec{v}_{\texttt{id}} \leftarrow \texttt{Dec}_{\texttt{sk}}(\vec{c'})$.
- Await a set C of indices of honest senders whose messages $\mathcal{S}$ wants to corrupt, and send $(\vec{v}_{\texttt{id}}, \texttt{id})$ to the receiver (for a chosen k), where $\vec{v}_{\texttt{id}}$ may be $\perp$ if decryption failed or if $\texttt{id} \in C$.

If the receiver is corrupt and there is at least one honest sender, await a message $(\texttt{sid}, \texttt{query}, \vec{c})$ from $\mathcal{S}$ on behalf of the receiver and then for each honest sender with identity `id`, do the following:

- Await a matrix $M_{\texttt{id}}$ from the sender.
- Await a ciphertext $\vec{c_{\texttt{id}}}$ from $\mathcal{S}$.
- Compute $\vec{c'_{\texttt{id}}} \leftarrow M_{\texttt{id}} \cdot \vec{c}$.
- Send $\vec{c'_{\texttt{id}}}$ to $\mathcal{S}$.

Figure 5. Functionality $\mathcal{F}^{\kappa}_{\mathsf{Mv}}$

in Figure 6.

Proposition 1. *The protocol Π_{Mv} UC-securely realises $\mathcal{F}^{\kappa}_{Mv}$ with abort, with computational security κ, in the $\mathcal{F}_{PubSub}$-hybrid model.*

Proof. To prove the theorem, we must show that for any adversary $\mathcal{A}$ against Π_{Mv}, there is a simulator $\mathcal{S}$ against $\mathcal{F}^{\kappa}_{\mathsf{Mv}}$ such that the view of the environment $\mathcal{Z}$ is indistinguishable between the two cases. We construct the simulator as follows. The simulator executes a copy of $\mathcal{F}_{\mathsf{PubSub}}$, runs the **Initialise** procedure, and responds to future queries as the functionality would.

Corrupt receiver If the receiver is corrupt, the simulator acts as a relay between $\mathcal{A}$ and $\mathcal{F}^{\kappa}_{\mathsf{Mv}}$. If there is at least one honest sender, $\mathcal{S}$ proceeds as follows. The simulator awaits some ciphertext $\vec{c}$ from $\mathcal{A}$ in the call $(\texttt{sid}, \texttt{publish}, (\vec{c} \| \texttt{qid}), \texttt{PUBLIC})$ to $\mathcal{F}_{\mathsf{PubSub}}$ on behalf of the corrupt receiver. The simulator forwards $\vec{c}$ to $\mathcal{F}^{\kappa}_{\mathsf{Mv}}$ via the message $(\texttt{sid}, \texttt{query}, \vec{c})$. and sends the message $(\texttt{sid}, \texttt{new\_message}, (\vec{c} \| \texttt{qid}), \texttt{PUBLIC})$ to $\mathcal{S}$ if there are any corrupt senders, as $\mathcal{F}_{\mathsf{PubSub}}$ would. For each honest sender, indexed by `id`, the simulator receives back one ciphertext $\vec{c'_{\texttt{id}}}$ from $\mathcal{F}^{\kappa}_{\mathsf{Mv}}$, samples $\texttt{rid}_{\texttt{id}}$ as an honest sender would, and sends the message $(\texttt{sid}, \texttt{publish}, (\vec{c}, \texttt{rid}_{\texttt{id}}), \texttt{qid})$ to its copy of $\mathcal{F}_{\mathsf{PubSub}}$, sending the resulting messages to $\mathcal{A}$ as in an honest execution.

Honest receiver If the receiver is honest and any sender is corrupt, the simulator awaits $\vec{c}$ from $\mathcal{F}^{\kappa}_{\mathsf{Mv}}$. (If no senders are corrupt, there is nothing to simulate.) The simulator samples `qid` as an honest receiver would and sends the message $(\texttt{sid}, \texttt{publish}, (\vec{c} \| \texttt{qid}), \texttt{PUBLIC})$ to its copy of $\mathcal{F}_{\mathsf{PubSub}}$ and sends the resulting messages to $\mathcal{A}$ as in an honest execution. On receipt of a message $(\texttt{sid}, \texttt{publish}, (\vec{c'} \| \texttt{rid}), \texttt{qid})$ from $\mathcal{A}$ to $\mathcal{F}_{\mathsf{PubSub}}$, the simulator sends $\vec{c'}$ to $\mathcal{F}^{\kappa}_{\mathsf{Mv}}$. If $\mathcal{S}$ does not receive a message on behalf of any corrupt sender, $\mathcal{S}$ sends the corresponding party sender indices to $\mathcal{F}^{\kappa}_{\mathsf{Mv}}$ as the set C.

Π_{Mv} in the $\mathcal{F}_{\mathsf{PubSub}}$-hybrid model

Initialise The senders and the receiver do the following:

- Send the message $(\texttt{sid}, \texttt{init})$ to $\mathcal{F}_{\mathsf{PubSub}}$.
- Subscribe to some public channel to which queries will be published by sending the command $(\texttt{sid}, \texttt{subscribe}, \texttt{PUBLIC})$ to $\mathcal{F}_{\mathsf{PubSub}}$.

Query The senders and receiver do the following:

- The receiver samples a new query ID $\texttt{qid}$ and subscribes to the relevant channel by sending the message $(\texttt{sid}, \texttt{subscribe}, \texttt{qid})$ to $\mathcal{F}_{\mathsf{PubSub}}$.
- The receiver with input vector $\vec{v}$ samples a new key pair $(\texttt{pk}, \texttt{sk}) \leftarrow \texttt{KeyGen}(\kappa)$.
- The receiver computes $\vec{c} \leftarrow \texttt{Enc}_{\texttt{pk}}(\vec{v})$ and sends the message $(\texttt{sid}, \texttt{publish}, (\vec{c} \| \texttt{qid}), \texttt{PUBLIC})$ to $\mathcal{F}_{\mathsf{PubSub}}$.
- The senders await the message $(\texttt{sid}, \texttt{new\_message}, (\vec{c} \| \texttt{qid}), \texttt{PUBLIC})$ from $\mathcal{F}_{\mathsf{PubSub}}$ and sample a response ID $\texttt{rid}$.
- The senders compute $\vec{c'} \leftarrow M \cdot \vec{c}$ using their input M, and send the message $(\texttt{sid}, \texttt{publish}, (\vec{c} \| \texttt{rid}), \texttt{qid})$ to $\mathcal{F}_{\mathsf{PubSub}}$.
- On receipt of any message $(\texttt{sid}, \texttt{new\_message}, (\vec{c} \| \texttt{rid}), \texttt{qid})$, the receiver computes $\vec{v'} \leftarrow \texttt{Dec}_{\texttt{sk}}(\vec{c})$ and (locally) returns $\vec{v'}$ or $\perp$ if decryption fails.

Figure 6. Protocol Π_{Mv}

Indistinguishability We now argue that the view of the environment interacting with the simulator and the ideal world instance is computationally indistinguishable from the view when performing Π_{Mv}. First, note that since $\mathcal{F}_{\mathsf{PubSub}}$ is honestly simulated, there is no way to distinguish between worlds using its behaviour.

When the receiver is corrupt, the simulator simply relays messages between the adversary and $\mathcal{F}_{\mathsf{Mv}}^{\kappa}$, so the distributions are identical in both worlds. When the receiver is honest, computations of the simulator and functionality together are the same as what an honest receiver does in Π_{Mv}, so the revealed ciphertexts are identically distributed in this case too. Note that the ciphertext is leaked to the adversary regardless of whether any parties have been (adaptively) corrupted by this point in the execution. This is because the ciphertext is published publicly using $\mathcal{F}_{\mathsf{PubSub}}$. This allows us to simulate even adaptive corruptions, where the adversary does not initially corrupt any sender. □

To instantiate our final protocol using $\mathcal{F}_{\mathsf{Mv}}^{\kappa}$, receivers (queriers) and senders (data owners) use a vision-language model to generate embedding vectors for the query and datasets as their inputs. When the querier has received a vector of inner products as the output, they can request data items corresponding to the indices of close matches (where they decide for themselves how close a 'successful' match is, and how many items to request).

7.4. Privacy Leakage

The UC proof shows that Π_{Mv} securely realises $\mathcal{F}_{\mathsf{Mv}}^{\kappa}$; we now briefly clarify what each party observes in the protocol to demonstrate that Π_{Mv} combined with the matrix row permutation described in Sec. 4.4 ensures the client cannot learn anything about the data owner's dataset.

Note that once a client has received an item from the dataset, they can freely recompute the embedding vector if they choose (if an open-weight VLM was used); embedding vector privacy is not required at this point since a corrupt client now knows the data item itself and the data owner has already 'sold' their content.

First, we consider the privacy of the client from the data owner. The data owner only observes the client's ciphertexts in the protocol, and receives a final set of indices for items the clients has selected. In our protocol, the client uses one SHE key per 'session' (query), and this key is used a number of times that is logarithmic in the size of the dataset. As such, the data owner is extremely limited in types of attack they can mount on the encryption scheme to determine the client's input vector (e.g. selective failure). An adversary could pretend to be multiple data owners to increase the number of ciphertexts they can test, but data owners do not know if a client has ceased computation because they have not found any 'reasonably close' match, because they received a ciphertext they could not decrypt, or because they simply chose to abort the protocol. Moreover, if it was found in a real deployment that many clients were receiving ciphertexts they could not decrypt, disincentives such

as nominal costs for posting and receiving messages could be added. Another mitigation could be for clients to respond occasionally even if they cannot decrypt data owner responses to hide decryption failure. The need for these mitigations depends on what behaviours are observed in a real deployment. Adding ZKPs for CKKS (as discussed in the paper body) could be a low-effort way to prevent these kinds of attack in a future iteration of PRISM.

We now consider the privacy of the data owner from the client. The client submits a ciphertext $\vec{c}$ of an embedding vector $\vec{a} \in \mathbb{F}^n$ and receives back a vector $\vec{c'}$ from a data owner. After decryption, the client learns a vector $\vec{b} \in \mathbb{F}^m$, which can be viewed as the inner product of their input vector $\vec{a}$ with each row of the data owner's matrix $M \in \mathbb{F}^{m \times n}$. If the client submits several linearly independent vectors $(\vec{a_i})_{i=1}^m$, then they can theoretically learn some subspace of the rowspace of M (i.e. the embedding space of their inputs), and possibly partially reverse-engineer the data owner's dataset as described in Sec. 4.4.

To mitigate this issue, we have prescribed that the data owner permute the rows of the matrix on each invocation of $\mathcal{F}^{\kappa}_{\mathsf{Mv}}$. We also require that M be padded with random vectors up to the ciphertext packing parameter ($4,096$ in our implementation) so this permutation is always meaningful.

The best strategy for the client would appear to be to submit low Hamming-weight vectors to learn small linear combinations of columns of M and from these attempt to reconstruct embedding vectors. Supposing the client submits standard basis vectors (e.g. $(0, 0, 1, 0, 0, \ldots, 0)$ to obtain columns of M, there are $4,096$ choose 512 (i.e. approximately 3.16×10^{668}) vectors to construct to find 4096 embedding vectors. While embedding vectors are clearly readily distinguishable from random vectors, we believe the task of determining embedding vectors from these known values to be suboptimal compared to

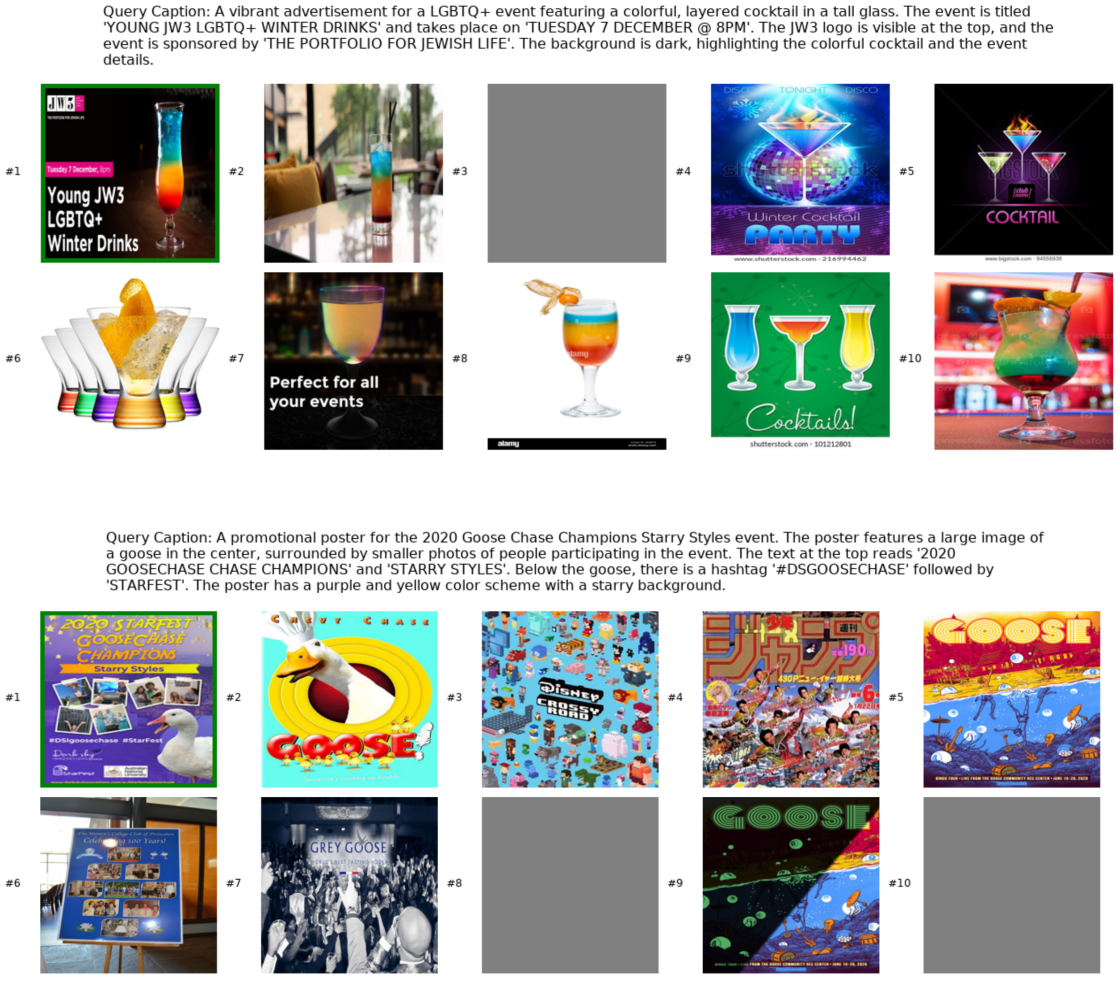

Figure 7. Examples of **correct retrievals** where the top-ranked result matches the ground truth image for the given query. *Note:* Gray images indicate items that were correctly retrieved by the index but were no longer available at their source URL at the time of visualization (link rot).

Figure 8. Examples of **failed retrievals** where the system did not return the exact ground truth image. Despite being classified as incorrect, the retrieved images often exhibit high semantic relevance to the text query, highlighting the robustness of the fuzzy search capability.

non-cryptographic ways to attack the system (e.g. spoofing, human factors, etc.). There may be other much better attack strategies, but it is not clear to the authors what these might be as the dimensions of the vectors involved are considerable.

7.5. Qualitative Results

In this section, we provide visual examples of the system's retrieval performance on the Recap-DataComp-1B dataset. Figure 7 demonstrates successful exact matches, while Figure 8 illustrates cases where the exact ground truth was missed, but the retrieved content remains semantically aligned with the query. Note: Gray images indicate items that were correctly retrieved by the index but were no longer available at their source URL at the time of visualization (link rot).

References

[1] *The Cambridge Handbook of Law and Policy for NFTs*. Cambridge University Press, 2024. 2

[2] Apple. CSAM detection technical summary. Technical summary, 2021. 2

[3] Benny Applebaum, Ivan Damgård, Yuval Ishai, Michael Nielsen, and Lior Zichron. Secure arithmetic computation with constant computational overhead. In *Annual International Cryptology Conference*. Springer, 2017. 10

[4] Muhammad Junaid Awan, Kar Balan, and John Collomosse. Decentralized creative copyright exchange in the age of generative AI. In *SIGGRAPH European Conference on Visual Media Production Demos Programme*, London, UK, 2025. ACM. 1

[5] Kar Balan, Shruti Agarwal, Simon Jenni, Andy Parsons, Andrew Gilbert, and John Collomosse. Ekila: Synthetic media provenance and attribution for generative art. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 913–922, Los Alamitos, CA, USA, 2023. IEEE Computer Society. 1

[6] Kar Balan, Andrew Gilbert, and John Collomosse. Content arcs: decentralized content rights in the age of generative ai. In *International Conference on AI and the Digital Economy (CADE 2025)*, 2025. 1

[7] Tadas Baltrušaitis, Chaitanya Ahuja, and Louis-Philippe Morency. Multimodal machine learning: A survey and taxonomy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018. 3

[8] Jean-Philippe Bossuat, Rosario Cammarota, Ilaria Chillotti, Benjamin R Curtis, Wei Dai, Huijing Gong, Erin Hales, Duhyeong Kim, Bryan Kumara, Changmin Lee, et al. Security guidelines for implementing homomorphic encryption. *Cryptology ePrint Archive*, 2024. 10

[9] Zvika Brakerski, Craig Gentry, and Shai Halevi. Packed ciphertexts in lwe-based homomorphic encryption. In *International Workshop on Public Key Cryptography*. Springer, 2013. 2, 3

[10] Uwe Breidenbach, Martin Steinebach, and Huajian Liu. Privacy-enhanced robust image hashing with bloom filters. In *Proceedings of the 15th International Conference on Availability, Reliability and Security*, New York, NY, USA, 2020. Association for Computing Machinery. 2

[11] Tu Bui, Daniel Cooper, John Collomosse, Mark Bell, Alex Green, John Sheridan, Jez Higgins, Arindra Das, Jez Keller, Oliver Thereaux, and Alan Brown. ARCHANGEL: Tamper-proofing Video Archives using Temporal Content Hashes on the Blockchain. In *CVPR Workshops (Computer Vision, AI and Blockchain), 2019*, 2019. 2

[12] Tu Bui, Daniel Cooper, John Collomosse, Mark Bell, Alex Green, John Sheridan, Jez Higgins, Arindra Das, Jared Keller, and Oliver Thereaux. Tamper-proofing Video with Hierarchical Attention Autoencoder Hashing on Blockchain. *IEEE Transactions on Multimedia (TMM), 2020*, 2020. 2

[13] Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *Proceedings 42nd IEEE Symposium on Foundations of Computer Science*. IEEE, 2001. 3, 9

[14] Ran Canetti, Palak Jain, Marika Swanberg, and Mayank Varia. Universally composable end-to-end secure messaging. In *Annual International Cryptology Conference*. Springer, 2022. 9

[15] Anrin Chakraborti, Giulia Fanti, and Michael K Reiter. Distance-aware private set intersection. In *32nd USENIX Security Symposium*, 2023. 10

[16] David L. Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Commun. ACM*, 24(2), 1981. 4

[17] Yunhao Chen, Shujie Wang, Xin Wang, and Xingjun Ma. Leakyclip: Extracting training data from clip. *arXiv preprint arXiv:2508.00756*, 2025. Computer Science - Cryptography and Security. 2, 5

[18] Jung Hee Cheon, Andrey Kim, Miran Kim, and Yongsoo Song. Homomorphic encryption for arithmetic of approximate numbers. In *International conference on the theory and application of cryptology and information security*, pages 409–437. Springer, 2017. 3

[19] Coalition for Content Provenance and Authenticity (C2PA). *C2PA Specification, Version 2.2*. Coalition for Content Provenance and Authenticity (C2PA), 2025. Version 2.2. 1

[20] John Collomosse and Andy Parsons. To Authenticity, and Beyond! Building safe and fair generative ai upon the three pillars of provenance. *IEEE Computer Graphics and Applications*, 44(03):82–90, 2024. 1

[21] CoSTAR National Lab, DECaDE, and Sheridans. Time to ACCCT: Providing creative industries and AI developers with a copyright framework of access, control, consent, compensation and transparency. Technical report, CoSTAR National Lab, 2025. 1

[22] Daniel Csizmadia, Andrei Codreanu, Victor Sim, Vighnesh Prabhu, Michael Lu, Kevin Zhu, Sean O'Brien, and Vasu Sharma. Distill clip (dclip): Enhancing image-text retrieval via cross-modal transformer distillation. *arXiv preprint arXiv:2505.21549*, 2025. 6

[23] Ivan Damgård, Marcel Keller, Enrique Larraia, Valerio Pastro, Peter Scholl, and Nigel P Smart. Practical covertly secure mpc for dishonest majority–or: breaking the spdz limits. Springer, 2013. 10

[24] Ivan Damgård, Valerio Pastro, Nigel Smart, and Sarah Zakarias. Multiparty computation from somewhat homomorphic encryption. In *Proceedings of the 32nd Annual Cryptology Conference on Advances in Cryptology — CRYPTO 2012 - Volume 7417*. Springer-Verlag, 2012. 10

[25] Inderjit S. Dhillon and Dharmendra S. Modha. Concept decompositions for large sparse text data using clustering. *Machine Learning*, 2001. 4

[26] Matthijs Douze, Alexandr Guzhva, Chengqi Deng, Jeff Johnson, Gergely Szilvasy, Pierre-Emmanuel Mazaré, Maria Lomeli, Lucas Hosseini, and Hervé Jégou. The faiss library. *IEEE Transactions on Big Data*, 2025. 4

[27] Benjamin Elizalde, Soham Deshmukh, and Huaming Wang. Natural language supervision for general-purpose audio representations. In *ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing*, 2024. 2

[28] Mark Everingham, Luc Van Gool, Christopher K I Williams, John Winn, and Andrew Zisserman. The PASCAL Visual Object Classes (VOC) challenge. *International Journal of Computer Vision*, 88(2), 2009. 5

[29] Thibauld Feneuil and Matthieu Rivain. SmallWood: Hash-based polynomial commitments and zero-knowledge arguments for relatively small instances. Cryptology ePrint Archive, Paper 2025/1085, 2025. 10

[30] Will Gottsegen. The nft market is already centralized. *CoinDesk*, 2021. Published October 2021. Updated May 2023. 2

[31] Grand View Research. Creator economy market size, share & trends analysis report by end use, by platform type, by creative service, by revenue channel, by region, and segment forecasts, 2025 - 2033. Technical Report GVR-4-68040-695-7, Grand View Research, 2024. 1

[32] R. Iannella and V. Rodríguez-Doncel. ODRL vocabulary & expression 2.2. W3C Recommendation, 2018. 1

[33] IETF HTTPBIS Working Group. Oblivious http (OHTTP). https://datatracker.ietf.org/doc/html/draft-ietf-ohai-ohttp, 2025. IETF draft specifying Oblivious HTTP to decouple client identity from request contents. 4

[34] Andrej Karpathy. Image captioning split for ms-coco and flickr30k. https://github.com/karpathy/neuraltalk2#coco, 2015. 5

[35] Hamid Kazemi, Atoosa Chegini, Jonas Geiping, Soheil Feizi, and Tom Goldstein. What do we learn from inverting clip models? *arXiv preprint arXiv:2403.02580*, 2024. 2

[36] Jaehoon Kim and Byoung Chul Ko. Scene graph and natural language-based semantic image retrieval using vision sensor data. *Sensors (Basel, Switzerland)*, 25(11), 2025. 2

[37] Byoung Chul Ko, JiHyeon Lee, and Jae-Yeal Nam. Automatic medical image annotation and keyword-based image retrieval using relevance feedback. *Journal of Digital Imaging*, 25(4), 2012. 2

[38] Zhengfeng Lai, Haotian Zhang, Bowen Zhang, Wentao Wu, Haoping Bai, Aleksei Timofeev, Xianzhi Du, Zhe Gan, Jiulong Shan, Chen-Nee Chuah, Yinfei Yang, and Meng Cao. Veclip: Improving clip training vianbsp;visual-enriched captions. In *Computer Vision – ECCV 2024: 18th European Conference, Milan, Italy, September 29–October 4, 2024, Proceedings, Part XLII*. Springer-Verlag, 2024. 3

[39] Xianhang Li, Haoqin Tu, Mude Hui, Zeyu Wang, Bingchen Zhao, Junfei Xiao, Sucheng Ren, Jieru Mei, Qing Liu, Huangjie Zheng, Yuyin Zhou, and Cihang Xie. What if we recaption billions of web images with llama-3? In *Proceedings of the 42nd International Conference on Machine Learning*, 2025. 5

[40] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft coco: Common objects in context. In *Computer Vision – ECCV 2014*. Springer International Publishing, 2014. 5

[41] Essam Mansour, Andrei Vlad Sambra, Sandro Hawke, Maged Zereba, Sarven Capadisli, Abdurrahman Ghanem, Ashraf Aboulnaga, and Tim Berners-Lee. A demonstration of the solid platform for social web applications. In *Proceedings of the 25th International Conference Companion on World Wide Web*. International World Wide Web Conferences Steering Committee, 2016. 2

[42] Ocean Protocol Foundation. Ocean protocol: Tools for the web3 data economy – technical whitepaper. https://oceanprotocol.com/tech-whitepaper.pdf, 2020. 2

[43] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *Proceedings of the 38th International Conference on Machine Learning*, 2021. 2, 3

[44] Mohamed Ragab, Yury Savateev, Reza Moosaei, Thanassis Tiropanis, Alexandra Poulovassilis, Adriane Chapman, and George Roussos. Espresso: A framework for empowering search on decentralized web. In *Web Information Systems Engineering – WISE 2023*. Springer Nature Singapore, 2023. 2

[45] Cyrus Rashtchian, Peter Young, Micah Hodosh, and Julia Hockenmaier. Collecting image annotations using amazon's mechanical turk. In *Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*. Association for Computational Linguistics, 2010. 5

[46] J. Shashank, P. Kowshik, Kannan Srinathan, and C.V. Jawahar. Private content based image retrieval. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*, 2008. 2

[47] Shutterstock. Content publishing standards: Contextual metadata. https://support.submit.shutterstock.com/s/article/Content-Publishing-Standards-Contextual-Metadata, 2025. 2

[48] Martin Steinebach, Sebastian Lutz, and Huajian Liu. Privacy and robust hashes. In *Proceedings of the 14th International Conference on Availability, Reliability and Security*, New York, NY, USA, 2019. Association for Computing Machinery. 2

[49] Oskar Thorén, Sanaz Taheri-Boshrooyeh, and Hanno Cornelius. Waku: A family of modular p2p protocols for secure censorship-resistant communication. In *2022 IEEE 42nd International Conference on Distributed Computing Systems Workshops (ICDCSW)*, 2022. 4

[50] Jiapeng Wang, Chengyu Wang, Kunzhe Huang, Jun Huang, and Lianwen Jin. Videoclip-xl: Advancing long description understanding for video clip models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, 2024. 2

[51] William Zerong Wang and Dongfang Zhao. Balancing privacy and efficiency: Music information retrieval via additive homomorphic encryption, 2025. 2

[52] Peter Young, Alice Lai, Micah Hodosh, and Julia Hockenmaier. From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions. *Transactions of the Association for Computational Linguistics*, 2, 2014. 5

[53] Dongfang Zhao. A note on efficient privacy-preserving similarity search for encrypted vectors. *arXiv preprint arXiv:2502.14291*, 2025. 2