Data Leakage in Visual Datasets

Patrick Ramos* Ryan Ramos* Noa Garcia
The University of Osaka
{patrickramos@is., ryanramos@is., noagarcia@}ids.osaka-u.ac.jp

Abstract

We analyze data leakage in visual datasets. Data leakage refers to images in evaluation benchmarks that have been seen during training, compromising fair model evaluation. Given that large-scale datasets are often sourced from the internet, where many computer vision benchmarks are publicly available, our efforts are focused into identifying and studying this phenomenon. We characterize visual leakage into different types according to its modality, coverage, and degree. By applying image retrieval techniques, we unequivocally show that all the analyzed datasets present some form of leakage, and that all types of leakage, from severe instances to more subtle cases, compromise the reliability of model evaluation in downstream tasks.

1. Introduction

Visual datasets are at the core of advancements in computer vision, serving not only as the foremost resources for model training but also as benchmarks for evaluating technological progress. Datasets have been central to the key milestones in the field, such as ImageNet [16] for image recognition, COCO [32] for object detection, and LAION [46] for image generation. Despite their impact, best practices for the creation and use of visual datasets have received little attention. As the demand for larger datasets keeps growing, analyzing their content has become a major challenge. In this context, audits have been crucial for revealing critical issues such as bias [21, 36], toxicity [6–8], or duplication [53], highlighting the urgent need for thorough dataset analysis.

In this paper, we explore an often-overlooked issue in large visual datasets: data leakage. Data leakage occurs when a model has access to some or all of the evaluation test data during training. This may result in inflated performance metrics and compromise the integrity of model evaluation — one of the fundamental principles in machine learning. In the era of datasets scraped from the internet [20, 46, 47], where most benchmarks for model evaluation are also publicly available online, the problem of data leakage stands out as particularly relevant. A potential consequence is that as large vision and language models (VLMs) [14, 31, 40] are trained on huge pre-training datasets from the web, any existing leakage can result in overly optimistic evaluations, making comparisons to models trained on datasets without leakage unfair.

Data leakage is an active research topic in the context of large language models (LLMs) [1, 3, 25, 53], as LLMs are trained on vast amounts of text from the internet, often including partial or complete portions of evaluation benchmarks [41]. While the natural language processing (NLP) community is actively working to detect and address data leakage [4, 15, 17, 19, 23, 34, 38, 39, 43, 45, 49],[1] there has been comparatively little focus on identifying overlaps of images in visual datasets. To close this gap, we analyze a variety of standard visual datasets and explore whether there are overlaps in test and training images. We do so by categorizing datasets into three types according to their standard use: PRETRAINING, *i.e.* datasets used for training large models, TRAINING, *i.e.* datasets for fine-tunning or training smaller models, and BENCHMARK, *i.e.* datasets for reporting models' performance.

We define data leakage across three dimensions: modality, coverage, and degree. *Modality* refers to the type of data being leaked, such as images with or without annotations. *Coverage* describes the relationship between the training and the evaluation splits, such as from the same or different dataset. Finally, *degree* specifies the level of similarity required for two images to be considered leaked, such as identical or near-identical. Each scenario of data leakage is described by a unique combination of these three dimensions. In all cases, data leakage is detected through image retrieval by extracting image representations and conducting an efficient k-nearest neighbor (knn) search.

Our method and leakage definitions are validated by experimental results in Sec. 6 and extensive qualitative examples are provided in the supplementary material. Fol-

*Equal contribution.

[1]There was a dedicated workshop on the topic at ACL 2024: the 1st Workshop on Data Contamination, https://www.aclweb.org/portal/content/first-workshop-data-contamination.

lowing our approach, we conduct a comprehensive leakage analysis in Sec. 7, covering 20 data splits across 7 popular visual datasets. We study both intra-dataset leakage, *i.e*., image overlap between the test and training splits of the same dataset, and inter-dataset leakage, *i.e*., image overlap between two different datasets. Additionally, we demonstrate the impact that leaked samples have on the evaluation of three downstream tasks: zero-shot image classification, supervised image classification, and image-to-text retrieval. Our main findings are:

- For all the datasets under analysis, data leakage is present within splits of the same dataset, *i.e*. training and tests splits of the same dataset have shared images.
- For all BENCHMARK under analysis, we identify instances of data leakage in the PRETRAINING datasets.
- *Hard* leakage rates (*i.e*. identical images) tend to be below 3%, whereas *soft* leakage (*i.e*. near-identical images) reaches rates of 7%. In total, we find up to 10% leakage.
- Both hard and soft leakage have a pronounced effect on the evaluation of downstream tasks. While this generally results in inflated performance metrics, we also observe instances of decreased performance when labels differ between training and testing.

Our results consistently show that models are capable of memorizing not only identical leaked samples but also near-identical ones. This poses a great risk to fair model evaluation, especially in the context of zero-shot tasks where models have been trained on different datasets with potentially different leakage rates.

2. Related work

Data leakage in visual datasets is often referred to as image duplication.[2] Several models that rely on vast amounts of internet-sourced images for training have adopted different techniques to detect leakage between their training and test sets [14, 35, 40, 59]. For instance, using RMAC features [52], Mahajan *et al*. [35] reported less than 1% overlap between their dataset collected from Instagram[3] and four standard validation sets: ImageNet [16], CUB [55], Places [60], and COCO [32]. On the other hand, in CLIP [40], Radford *et al*. trained a custom duplication detector and found substantial overlap with rates reaching up to 21.5% in all but synthetic datasets (MNIST [30], CLEVR [26]) and those created post-training cutoff (ObjectNet [5], Hateful Memes [28]). Alternatively, in OpenCLIP [14], pHash [58] was used to estimate the leakage between LAION-400M [46] and the downstream datasets, ranging from 1% to 5%.

[2] *Duplication* can refer to the presence of identical images within the same dataset split, such as multiple copies of the same image in the training set. This scenario typically does not pose a significant problem. Therefore, we use the term *leakage* to specifically describe instances where duplicates occur between a training and a test set.

[3] https://www.instagram.com/

Leakage in visual datasets has been primarily conducted by the authors of models themselves, often concluding that there is a minimal impact on the evaluations. However, as large datasets and models are later applied to a wide range of tasks beyond their original scope, we believe an independent and systematic assessment is essential.

3. Data leakage definition

Data leakage compromises the integrity of a benchmark by using information from the evaluation split during the training process of a model. This can occur in different forms. To systematically characterize the issue, we break it down into three dimensions: *modality*, *coverage*, and *degree*.

Leakage modality is the dimension that defines the type of data being leaked. In visual datasets, we distinguish between two modalities:

- **Image-only leakage**: only the images from the evaluation set are exposed during training, while their corresponding annotations or labels remain unseen.
- **Full leakage**: both the images and their associated annotations or labels are exposed during training.

Leakage coverage is the dimension that defines the relationship between the training and evaluation splits of the leaked samples. We consider two scenarios:

- **Intra-dataset leakage**: occurs when there is an overlap between training and evaluation samples within the same dataset. This type of leakage compromises the integrity of the evaluation protocol for that specific dataset.
- **Inter-dataset leakage**: occurs when samples from an evaluation dataset are leaked into a different dataset used in the training process of a model. This type of leakage can affect the generalizability of the model across datasets, and it can be particularly problematic when benchmarking models trained on different datasets.

Leakage degree is the dimension that defines the level of similarity required between two images for them to be considered leaked. We distinguish between two cases:

- **Hard leakage**: occurs when identical images appear in both the training and evaluation sets. This represents a direct and unambiguous form of leakage, where the overlap between datasets is explicit and easily detectable.
- **Soft leakage**: occurs when nearly identical images are present in both the training and evaluation sets. Unlike hard leakage, soft leakage involves images with minor variations. Although soft leakage has been often overlooked, we show in Sec. 7.3 that it can greatly impact the evaluation of model generalization.

Each leakage scenario is defined by a unique combination of attributes across the three dimensions. For example, an image-only, intra-dataset, and hard leakage scenario describes a situation where the leakage occurs exclusively through exact images within the same dataset.

4. Data leakage detection

To detect data leakage in visual datasets, we need to identify whether images from a test split have been leaked into a train split. We formulate the problem as an image retrieval task (Sec. 4.1) and categorize leakage degree into hard and soft by thresholding (Sec. 4.2).

4.1. Image retrieval

Given two datasets, one commonly used for training $\mathcal{T} = \{\boldsymbol{x}_o\}$ and the other for evaluation $\mathcal{E} = \{\boldsymbol{x}_q\}$, where $\boldsymbol{x}_o$ and $\boldsymbol{x}_q$ are images, the goal is to find how many images from $\mathcal{E}$ can be found in $\mathcal{T}$. Using image retrieval terminology, we treat images in $\mathcal{E}$ as *queries* and search for them within $\mathcal{T}$, referred to as the *collection*.

Each image $\boldsymbol{x}$, whether in the query or in the collection, is represented by features obtained from an image encoder $e(\cdot)$, yielding the encoded representations $\boldsymbol{c} = e(\boldsymbol{x}_o)$ and $\boldsymbol{q} = e(\boldsymbol{x}_q)$. These image representations must contain enough information for detecting leaked images while being computationally efficient to handle large-scale data and robust enough to account for small transformations between datasets, such as resizing and cropping. Given the size of web-scale datasets, directly extracting representations for every image is computationally expensive. When available, we use pre-computed image representations provided by dataset authors, typically pre-trained CLIP [40].

Depending on the size of the collection $|\mathcal{T}|$, image retrieval is conducted as:

- **direct search**: if $|\mathcal{T}|$ is sufficiently small, we match each query representation $\boldsymbol{q}$ with each representation in the collection $\boldsymbol{c}$ to obtain a similarity score per pair $s_{q,c} = \cos(\boldsymbol{q}, \boldsymbol{c})$, where cos is cosine similarity;
- **knn search**: for larger datasets, we conduct a knn search with Faiss [18] by building an index I with the representations in $\mathcal{T}$ and use it for fast search given $\boldsymbol{q}$. The search is conducted based on similarities $s_{q,c} = \text{faiss}_{\text{sim}}(\boldsymbol{q}, \boldsymbol{c}, \boldsymbol{I})$, where $\text{faiss}_{\text{sim}}$ returns the cosine similarity between $\boldsymbol{q}$ and the indexed representation of $\boldsymbol{c}$ in I.

4.2. Leakage degree

The leakage degree is the dimension that defines how similar two images need to be to classify them as leakage. We use the similarity score $s_{q,c}$ obtained through direct or knn search to identify whether a pair of images are identifcal (hard leakage) or near-identical (soft leakage).

For a given pair of images, they are considered hard leakage when the similarity score is above a threshold

$$s_{q,c} \geq \tau_h, \tag{1}$$

whereas if the similarity score falls within

$$\tau_s \leq s_{q,c} < \tau_h \tag{2}$$

Table 1. Datasets details by type.

dataset	split	images	source	type
coco	test2014	$40k$	flickr	BENCHMARK
coco	test2015	$81k$	flickr	BENCHMARK
coco	test2017	$40k$	flickr	BENCHMARK
flickr30k	all	$31k$	flickr	BENCHMARK
gcc	val	$13k$	flume	BENCHMARK
imagenet	test	$100k$	flickr	BENCHMARK
imagenet	val	$50k$	flickr	BENCHMARK
openimages	test	$125k$	flickr	BENCHMARK
openimages	val	$41k$	flickr	BENCHMARK
textcaps	test	$3k$	openimages	BENCHMARK
coco	val2014	$40k$	flickr	BENCHMARK TRAINING
coco	val2017	$5k$	flickr	BENCHMARK TRAINING
coco	train2014	$82k$	flickr	TRAINING
coco	train2017	$118k$	flickr	TRAINING
coco	unlabeled	$123k$	flickr	TRAINING
textcaps	train	$25k$	openimages	TRAINING
gcc	train	$2,874k$	flume	TRAINING PRETRAINING
imagenet	train	$1,281k$	internet	TRAINING PRETRAINING
openimages	train	$1,743k$	flickr	TRAINING PRETRAINING
laion	all	$407,314k$	commoncrawl	PRETRAINING

the pair is considered soft leakage. The hard and soft leakage rates, H and S respectively, are computed as

$$\text{H} = \frac{N_h}{|\mathcal{E}|}, \qquad \text{S} = \frac{N_s}{|\mathcal{E}|}, \tag{3}$$

where N_h and N_s are the number of hard and soft leakage samples found, respectively. The thresholds τ_s and τ_h are found empirically and validated in Sec. 6.2.

5. Experimental settings

Setup Our default image encoder, $e(\cdot)$, is a pretrained CLIP ViT-B/32 [40]. In the knn search, we use the AutoFaiss[4] implementation for Faiss.

Datasets We categorize datasets into three types. PRETRAINING are large datasets sourced from the Internet and used to train large models. TRAINING are standard sets, typically annotated, used for model training or finetuning. BENCHMARK are smaller, annotated datasets used for evaluation. As summarized in Tab. 1, our analysis comprises 20 data splits across 7 datasets:

COCO [32] contains several splits (train, val, test, unlabeled) across different versions (2014, 2015, 2017). The images were collected from Flickr[5], and each image includes object annotations and five captions. COCO is the standard dataset for object detection [42, 56] and image captioning [54]. The dataset is used in different ways in the literature. While some papers [2, 22, 33] report results on the test splits, which require evaluation via a server, others [9] use the validation split, and some [42] report results on both. Additionally, it is common to augment the training data with

[4] https://github.com/criteo/autofaiss
[5] https://www.flickr.com/

the validation set to improve performance [27]. We use the test and val splits as BENCHMARK, with the val also serving as TRAINING, along with the train and unlabeled splits.

Flickr30k [57] has a single split with 31,783 images collected from Flickr, where each image is annotated with a caption. Train, val, and test splits were later introduced in [27]. We use it as BENCHMARK.

GCC [48] contains about 3.3 million images from the internet collected through the crawler Flume [11]. Images are divided into train and val splits, and each image is paired with an alt-text caption. Due to broken links, we could only download 2,874,229 train and 13,354 val images. The val split is used to report results [12], making it a BENCHMARK. Meanwhile, the training split has been used for training VLMs [13, 51], so we use it as PRETRAINING.

ImageNet [16] has more than a million images divided into train, val, and test splits. Images are collected from the internet[6] and annotated with class labels. Given its widespread use for training visual models that serve as backbones for downstream tasks [10, 24], we use its training split as PRETRAINING. The test and val splits are used as BENCHMARK.

LAION [46] contains about 400 million images crawled with Common Crawl[7] from the internet. Each image is paired with an alt-text caption. LAION is designed for training large VLMs [14, 44], and does not provide data splits. We use the entire dataset as PRETRAINING.

OpenImages [29] has several versions. We use OpenImages v4, which contains about 2 million images from Flickr split into train, val, and test. The dataset is commonly used for image classification and object detection. We use the val and test splits as BENCHMARK and the training split as PRETRAINING.

TextCaps [50] is a subset of OpenImages with human annotated captions. Data is split into 3,353 test and 25,119 training images, which we use as BENCHMARK and TRAINING, respectively.

6. Method evaluation

We validate the proposed method by showing the efficacy of the image retrieval (Sec. 6.1) and inspecting the choice of thresholds in the leakage degree (Sec. 6.2).

6.1. Image retrieval evaluation

To be able to detect small variations in images such as cropping or scaling, we evaluate the data leakage detection method with a particular focus on the cases where images undergo non-semantic transformations. We conduct this evaluation on Flickr30k. We use the full dataset, *i.e.*, 31,783 images, as the collection, and we randomly choose 5,000 images as queries. Given a query image (with or without a non-semantic transformation), the goal is to retrieve the original image from the collection.

The evaluation flow is as follows: for images in the collection, we just extract their encoded representations and store them. For query images, we apply a non-semantic transformation before feeding them to the image encoder. Then, the encoded representation of a transformed query is matched against all the representations in the collection with direct search, and the collection image with the highest cosine similarity is retrieved. Accuracy is measured as recall at 1 (R@1), *i.e.*, the number of retrieved images corresponding to their original query over the number of queries.

Image encoders We evaluate three types of image encoders: *resnet*, *dino2*, and *clip*. Image representations from *resnet* are extracted from the second-to-last layer of a 152-layer ResNet [24] pre-trained on ImageNet. The *dino2* encoder is a DINOv2 ViT-B/14 [37], while *clip* is the pre-trained CLIP ViT-B/32 [40] image encoder.

Image transformations We use four types of non-semantic transformations:

- **Geometric**: vertical (*flip-v*) and horizontal (*flip-h*) flips, and D degrees rotations with $D = \{45, 135, 225, 315\}$, namely *rot-45*, *rot-135*, *rot-225*, and *rot-315*.
- **Cropping**: removes B pixels from all four sides of the image, with $B = \{20, 50, 100\}$, namely *crop-20*, *crop-50*, and *crop-100*.
- **Pixelization**: Gaussian filters (*gauss*), Gaussian noise (*noise*), and downsizing the image to S pixels, with $S = \{128, 256\}$ as *rs-128*, *rs-256*.
- **Color**: grayscale (*gray*), color inversion (*invert*), and red, green, and blue colorizations (*red*, *green*, *blue*).

Examples of the transformations can be found in the supplementary material.

Results Results are shown in Fig. 1, where each transformation is plotted along the circular axis, with the lines representing R@1 for the different image encoders. When query images are not transformed (*original*), the three encoders achieve a 100% performance. For transformed images, the *clip* encoder performs well in cropping, horizontal flipping, noise, and resizing, and it outperforms *resnet* in colorization and pixelation. The most robust encoder for colorization is *dino2*, probably due to its self-supervised training process. However, *dino2* underforms at geometric and cropping transformations, which are the most common type of transformation observed when images are duplicated across datasets. Given this and considering that the largest dataset in our analysis (*i.e.*, LAION) provides

[6] While the val and test images are from Flickr, the source of the train images is not specified.

[7] https://commoncrawl.org/

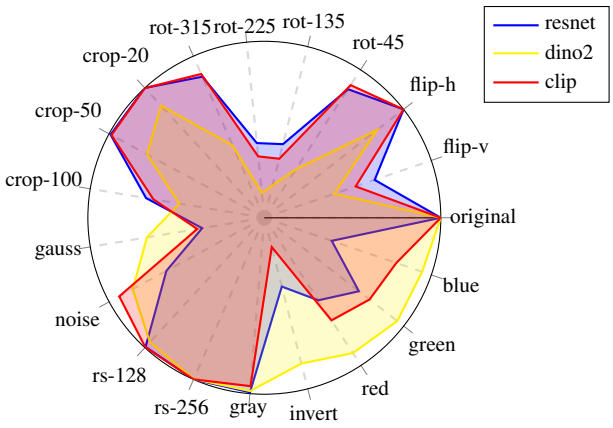


Figure 1. Leakage detection R@1 on $5,000$ query images from the Flickr30k dataset under several transformations.

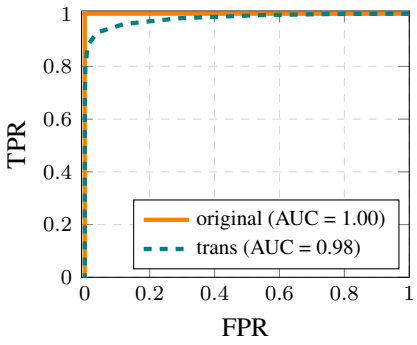


Figure 2. ROC curve across different thresholds on the Flickr30k dataset. For images without transformations (original), the retrieval achieves near-perfect performance with an AUC of almost 1.00, and even when images undergo transformations (trans), the performance remains high, with an AUC of 0.98.

pre-computed clip embeddings, *clip* stands out as our preferred choice, offering the best balance between robustness to transformations and computational efficiency.

6.2. Threshold choice

Next, we examine the definition of *hard* and *soft* leakage together with the choice of thresholds. Using the same experimental set-up as in Sec. 6.1, we compute the true positive rate (TPR) and false positive rate (FPR) for different thresholds under two conditions: when query images are not transformed (*original*) and when query images are transformed as in Sec. 6.1 (*trans*). The receiver operating characteristic (ROC) curves are plotted in Fig. 2. For the original case, the area under the curve (AUC) is near perfect,[8] while when query images suffer from some non-semantic transformation, the AUC is still remarkably high (0.98).

With respect to the leakage degree, we inspect the choice of hard and soft thresholds in detail:

- *Hard leakage* is defined as the leakage of identical images. We choose $\tau_h = 0.98$ as hard leakage threshold, which achieves a FPR of 0.0, and a TPR of 1.00 (original) and 0.08 (trans). This aligns precisely with the strict definition of hard leakage for identical images: no false positives occur, and true positives are detected only when the image is either identical (original) or undergoes nearly imperceptible transformations.
- *Soft leakage* is defined as the leakage of near-identical images. We choose $\tau_s = 0.95$ as soft leakage threshold, which results in a higher TPR of 1.00 (original) and 0.16 (trans), while the FPR remains extremely low at a maxiumum of 2.08×10^{-7}. Again, this is the expected behavior for *soft* leakage detection, where images that undergo some transformations and hence, are not exactly identical, can be detected while maintaining a very low false positive rate.

Our choice of thresholds prioritizes a low FPR over a high TPR to ensure that any detected image is truly a leaked image. As a result, our analysis and findings are conservative: there may be additional leaked images that are not detected. A qualitative analysis of the threshold selection is provided in the supplementary material.

7. Leakage analysis

Focusing on the image-only modality, we corroborate the existence of intra-dataset (Sec. 7.1) and inter-dataset (Sec. 7.2) leakage on several widely-used benchmarks. After that, we present a comprehensive analysis on the impact of leakage on the evaluation of downstream tasks, including how leakage modality and degree affects the overall performance of benchmarks (Sec. 7.3).

7.1. Intra-dataset leakage

To compute intra-dataset leakage (*i.e.*, image overlap between training and evaluation samples within the same dataset) we use datasets with both BENCHMARK and TRAINING / PRETRAINING splits. That leaves us with five datasets: COCO, GCC, ImageNet, OpenImages, and TextCaps. For COCO, test splits are compared against train, val, and unlabeled splits, and the val splits against the train and unlabeled splits. We note that the val2014 split is included in train2017.

Results Results are shown in Fig. 4. ImageNet test and val, GCC val, and COCO val 2017 are the most leaked datasets. The ImageNet test split has a hard leakage of 1.54% and a soft leakage of 1.95%, while the val split has a slightly higher hard leakage of 1.58% but a lower soft leakage of 1.78%. The second highest leaked dataset is GCC followed by COCO, in which the val2017 split has

[8] Computed AUC value of 0.99999999370713.

	Flickr30k	coco-test2014	coco-val2014	textcaps-test	openimages-test	openimages-val	imagenet-test	imagenet-val
GCC	0.02	0.03	0.03	0.03	0.06	0.05	0.03	0.03
OpenImages	0.80	1.46	1.55	0.03	0.05	0.05	0.15	0.20
ImageNet	0.44	0.54	0.46	0.57	0.32	0.32	1.54	1.58
LAION	0.72	0.46	0.56	3.01	2.90	2.83	0.62	0.54

(a) hard leakage

	Flickr30k	coco-test2014	coco-val2014	textcaps-test	openimages-test	openimages-val	imagenet-test	imagenet-val
GCC	0.02	0.33	0.28	0.42	0.71	0.71	0.33	0.30
OpenImages	0.15	0.69	0.63	0.84	1.36	1.38	0.47	0.49
ImageNet	0.06	0.38	0.40	0.45	0.66	0.67	1.95	1.78
LAION	1.09	4.13	3.71	6.59	7.77	7.91	4.02	4.02

(b) soft leakage

Figure 3. Inter-dataset leakage. Columns indicate BENCHMARK and rows PRETRAINING datasets. Left is for hard leakage (in red), and right for soft leakage (in blue).

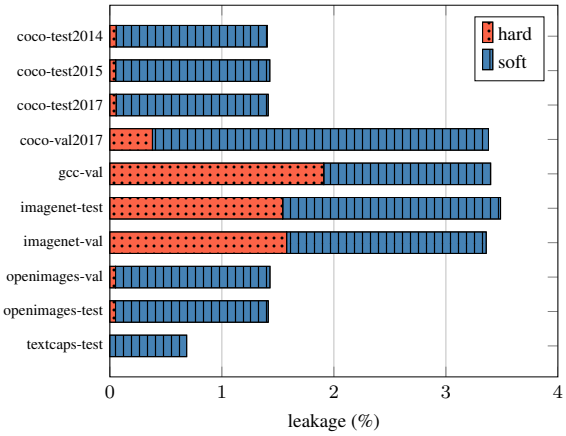


Figure 4. Intra-dataset leakage. Each dataset split in the y-axis is matched against the samples in their corresponding train split.

a soft leakage of 3%. COCO test splits also present a soft intra-dataset leakage between 1.35% and 1.38%. Both the test and val splits of OpenImages have small rates of hard leakage, 0.05%, but larger rates of soft leakage, 1.36% and 1.38%, respectively. TextCaps has the smallest leakage rate, 0.69%, and we do not find any hard leakage.

These results show that intra-dataset leakage occurs in *all* the analyzed datasets, either in its hard or soft degree. Given that data leakage can compromise model evaluation and that datasets are specifically designed for this purpose, intra-dataset leakage is a risk that *by design* should not exist. Yet, we have identified multiple instances in all datasets.

7.2. Inter-dataset leakage

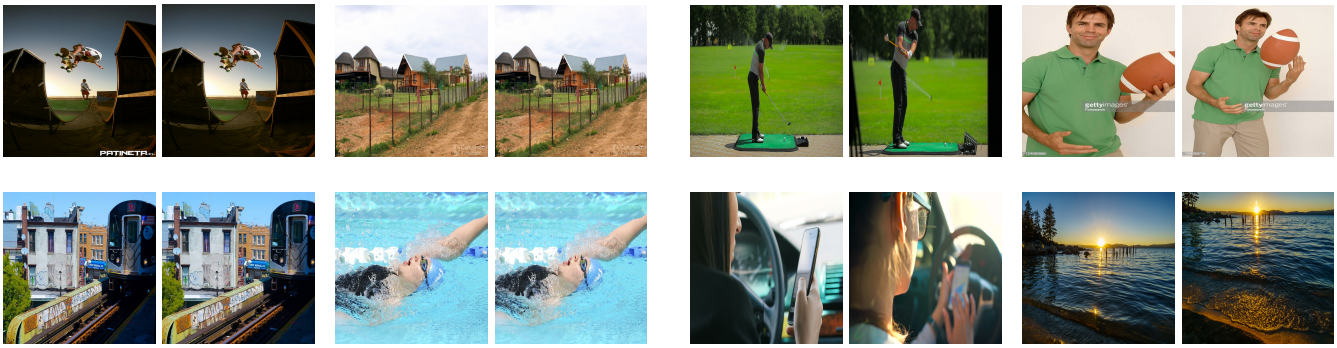
To compute intra-dataset leakage (*i.e.*, leakage between different datasets when a model has been trained on PRETRAINING and evaluated on a BENCHMARK), we use GCC train, ImageNet train, OpenImages train, and LAION as collection data and COCO 2014 test and val, Flickr30K, TextCaps test, OpenImages test and val, and ImageNet test and val as query data.

We extract CLIP ViT-B/32 embeddings in all datasets except LAION, in which we use the provided precomputed embeddings.[9] We found that LAION's precomputed embeddings do not fully match embeddings extracted by CLIP's official implementation.[10] This is addressed by rescaling query images as in the `clip-retrieval`[11] repository. We conduct a knn search by building a single index I per collection dataset, except for LAION, which we found that due to its size, a single index yielded poor retrieval. Instead, we create an index per data partition, with each partition containing a million images.

Results Results are shown in Fig. 3, where the columns are BENCHMARK used as queries and the rows are PRETRAINING used as collections. The left part, in red, reports the hard leakage rate, whereas the right part, in blue, is the soft leakage rate, both expressed as a percentage. Overall, leakage exists in all datasets.

Figure 3a shows that while most of the rates are relatively small, the highest rates of hard leakage are found in LAION, specifically for the TextCaps and OpenImages test splits, with about 3% hard leaked images (*i.e.*, identical images). OpenImages train split contains about 1.5% hard leaked images from COCO test and val splits. ImageNet, which is one of the most common datasets for pretraining, also contains hard leaked images from all the benchmarks. Among

[9] https://laion.ai/blog/laion-400-open-dataset/
[10] https://github.com/openai/CLIP
[11] https://github.com/rom1504/clip-retrieval



(a) hard leakage

(b) soft leakage

Figure 5. Examples of leaked images. Left: hard leakage, where images are identical, both within the same dataset (first row) and between datasets (second row). Right: soft leakage, where images are near-identical.

the PRETRAINING datasets, GCC has the least hard leakage, with all rates remaining below 1%. Soft leakage (*i.e.*, near-identical images) is shown in Fig. 3b. The dataset with the most soft leakage is LAION, ranging from 1.09% to 7.91%. When comparing different BENCHMARK, OpenImages test and val splits and TextCaps test split are the most leaked ones, while Flickr30k shows the least soft leakage rates. Examples of both hard and soft leakage are shown in Fig. 5. While hard-leaked images are identical, soft-leaked samples are also extremely similar, often depicting the same person or object in different poses.

In total, ImageNet, OpenImages, TextCaps, COCO, and Flickr30k have 4.64%, 10.67%, 9.60%, 4.59%, and 1.81% of their test samples leaked into LAION, respectively. Although leakage rates may represent only a small fraction of the BENCHMARK datasets, these results are worrisome, as a model can potentially memorize the features of the leaked test images and affect the downstream evaluation, as we will see in Sec. 7.3.

7.3. Impact on downstream evaluation

The next natural step is to study the impact of data leakage on downstream evaluations. In particular, we evaluate three tasks: zero-shot classification, supervised classification, and text-image retrieval. In each task, we evaluate the performance of a pretrained model on different subsets of a BENCHMARK in which we know which samples have been leaked into the PRETRAINING dataset. Then, we compare the results on different subsets of the benchmark:

- **original dataset**: the whole benchmark containing N samples.
- **leaked set**: a subset of A samples identified as leaked.
- **non-leaked set**: a subset of $N - A$ samples that have not been identified as leaked.
- **random set**: a subset of A samples randomly selected from the original dataset, serving as a control set.

Table 2. Zero-shot classification accuracy on ImageNet val split for different subsets, with and without leakage. The last column (gain) indicates the difference with respect to the original dataset (*i.e.*, first row). We highlight the rows of the leaked subsets.

subset	leakage	images	openclip	gain
original	-	50,000	54.18	-
non-leaked	hard	49,729	54.15	−0.03
leaked	hard	271	**60.15**	+5.97
random	-	271	53.51	−0.67
non-leaked	soft	2,281	53.54	−0.64
leaked	soft	2,281	**67.65**	+13.47
random	-	2,281	54.84	+0.66

Zero-shot classification As VLMs tend to not fully disclose their training data [40], it is usually impossible to identify their leaked samples. For this analysis, we rely on OpenCLIP ViT-B-32[12] pretrained on LAION, and evaluate it on the ImageNet val split following [40].

Accuracy results for the different subsets on hard and soft leakage are shown in Tab. 2. For both hard and soft leakage, the subset of leaked images consistently achieves much higher accuracy than the non-leaked images (+14.11 for soft leakage) and the randomly selected images (+12.81 for soft leakage). This strongly suggests that the model benefits from exposure to leaked images during training. Moreover, the performance of the non-leaked subset decreases from 54.15 to 53.54 when we exclude not only identical images but also near-identical images. This corroborates that near-identical images (*i.e.* soft leakage) are contributing to improved performance when seen during training. In a zero-shot scenario where models are trained on different PRETRAINING datasets, this is particularly problematic, as the number of leaked images may vary between models,

[12] https://github.com/mlfoundations/open_clip

Table 3. Supervised image classification accuracy on ImageNet val split for different subsets, with and without leakage. *Gain* columns indicate the difference with respect to the original dataset (*i.e.*, first row). We highlight the rows of the leaked subsets.

subset	leakage	images	resnet50	gain	resnet152	gain
original	-	50,000	80.37	-	82.83	-
non-leaked	hard	49,211	81.18	+0.81	83.65	+0.82
leaked	hard	789	30.16	−50.21	31.69	−51.14
↪ same label	hard	11	**100.00**	+19.63	**100.00**	+17.17
↪ different label	hard	778	29.18	−51.19	30.72	−52.11
random	-	789	83.27	+2.90	84.54	+1.71
non-leaked	soft	36,720	81.00	+0.63	83.52	+0.69
leaked	soft	1,680	62.44	−17.93	62.80	−20.03
↪ same label	soft	812	**96.06**	+15.06	**95.44**	+11.92
↪ different label	soft	868	30.99	−50.01	32.26	−51.26
random	-	1,680	80.24	−0.13	83.04	+0.21

Table 4. Image-to-text retrieval on Flickr30k for different subsets, with and without leakage. Leaked subsets are highlighted.

subset	leakage	R@1	R@5	R@10
original	-	33.22	55.25	64.13
non-leaked	hard	36.00 ± 3.14	58.65 ± 4.19	66.15 ± 2.96
leaked	hard	**45.55** ± 1.19	**70.80** ± 1.11	**79.20** ± 0.67
random	-	34.30 ± 2.85	55.95 ± 2.54	64.80 ± 2.67
non-leaked	soft	33.05 ± 3.56	54.95 ± 3.12	63.75 ± 2.15
leaked	soft	**34.90** ± 2.07	**59.05** ± 2.76	**68.25** ± 2.12
random	-	32.15 ± 3.56	56.35 ± 5.18	64.25 ± 4.21

making comparisons between them especially unfair.

Supervised classification We check how the intra-dataset leakage on ImageNet affects two standard backbones, ResNet50 and ResNet152, both pretrained on the train split and evaluated on different subsets of the val split.

The results are in Tab. 3. At first sight, it seems that hard leakage is greatly detrimental to model performance, reducing accuracy from 80.37% on the original dataset to just 30.16%. An in-depth inspection reveals the cause of this behavior: the labels. Unlike in the zero-shot task, where labels are not used during training, for supervised training, we observe full leakage. We find that 98.61% of the hard-leaked images have different labels in the train and val splits of ImageNet. In the samples in which the leaked labels are the same, the accuracy raises to 100%, while in the samples with different labels, accuracy is reduced to 29.18% and 30.72%, demonstrating that both ResNet50 and ResNet152 are able to memorize leaked data. This also occurs in the soft-leaked subset: leaked images where the label is the same get 96.06% and 95.44% accuracy, in contrast to those where it is different, with accuracies of 30.99% and 32.26%, respectively.

Image-to-text retrieval Other than image classification, we study the impact of data leakage on another task: image-to-text retrieval. Given a query image, the goal is to find its associated caption within a collection of captions. We evaluate OpenCLIP ViT-B-32 trained on LAION on different subsets of the Flickr30k. For the results to be comparable between subsets, we fix the number of queries to 200 images and the size of the collection to 31,783 captions. We repeat each experiment 10 times with 200 random images from each subset and report results as the mean and standard deviation of retrieval at k (R@k), $k = 1, 5, 10$.

Results are shown in Tab. 4. The best performance is achieved on the hard-leaked subset, being +9.55 and +13.05 better than the non-leaked subset on R@1 and R@10, respectively. Soft leakage has similar behavior, with soft-leaked images being +4.5 R@10 over the equivalent non-leaked images. While in this case most subsets outperform on average the full dataset results, this can be attributed to the use of a much smaller query size of 200 instead of the original 31,783 images in Flickr30k. Regardless, the non-leaked subsets tend to have a larger variance than the leaked subsets, meaning that the latter produce more stable results, probably due to the model being familiar with the leaked samples seen at training time.

8. Final remarks

Data leakage is a widespread issue, prevalent in most visual datasets. Leakage can obscure the generalization ability of models, which is particularly problematic when comparing models trained on different datasets, leading to unfair comparisons. We urge dataset designers to carefully consider the implications of these evaluations. For a fairer model evaluation, we recommend the use of duplicate detectors that considers both hard and soft leakage. Ideally, leaked images should be removed from the training set, and if not possible, they should at least be removed from the test set.

9. Conclusion

We thoroughly investigated data leakage in visual datasets. We started by characterizing visual leakage, and proposed a data leakage detector based on CLIP image retrieval. We demonstrated the presence of data leakage not only between different datasets (inter-dataset) but also within splits of the same dataset (intra-dataset). Furthermore, we analyzed the impact of data leakage on three downstream tasks, and showed that models not only do memorize identical samples (hard leakage), but also near-identical ones (soft leakage). Overall, we showed consistent evidence that leakage poses a serious threat to fair model evaluation in visual datasets, compromising one of the most fundamental machine learning principles: to not evaluate models on their training data.

Acknowledgments

This work was supported by JSPS KAKENHI No. JP23H00497 and JP22K12091.

References

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. GPT-4 technical report. Technical report, OpenAI, 2024. 1

[2] Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, et al. Flamingo: a visual language model for few-shot learning. In *NeurIPS*, 2022. 3

[3] Rohan Anil, Andrew M Dai, Orhan Firat, Melvin Johnson, Dmitry Lepikhin, Alexandre Passos, Siamak Shakeri, Emanuel Taropa, Paige Bailey, Zhifeng Chen, et al. PaLM 2 technical report. Technical report, Google, 2023. 1

[4] Simone Balloccu, Patrícia Schmidtová, Mateusz Lango, and Ondřej Dušek. Leak, cheat, repeat: Data contamination and evaluation malpractices in closed-source LLMs. In *ACL*, 2024. 1

[5] Andrei Barbu, David Mayo, Julian Alverio, William Luo, Christopher Wang, Dan Gutfreund, Josh Tenenbaum, and Boris Katz. ObjectNet: A large-scale bias-controlled dataset for pushing the limits of object recognition models. In *NeurIPS*, 2019. 2

[6] Abeba Birhane and Vinay Uday Prabhu. Large image datasets: A pyrrhic win for computer vision? In *WACV*, 2021. 1

[7] Abeba Birhane, Vinay Uday Prabhu, and Emmanuel Kahembwe. Multimodal datasets: misogyny, pornography, and malignant stereotypes, 2021.

[8] Abeba Birhane, Sanghyun Han, Vishnu Boddeti, Sasha Luccioni, et al. Into the LAION's den: Investigating hate in multimodal datasets. In *NeurIPS*, 2024. 1

[9] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with Transformers. In *ECCV*, 2020. 3

[10] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised Vision Transformers. In *ICCV*, 2021. 4

[11] Craig Chambers, Ashish Raniwala, Frances Perry, Stephen Adams, Robert R Henry, Robert Bradshaw, and Nathan Weizenbaum. FlumeJava: easy, efficient data-parallel pipelines. *ACM SIGPLAN Notices*, 2010. 4

[12] Tianwei Chen, Yusuke Hirota, Mayu Otani, Noa Garcia, and Yuta Nakashima. Would deep generative models amplify bias in future models? In *CVPR*, 2024. 4

[13] Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. UNITER: UNiversal Image-TExt Representation learning. In *ECCV*, 2020. 4

[14] Mehdi Cherti, Romain Beaumont, Ross Wightman, Mitchell Wortsman, Gabriel Ilharco, Cade Gordon, Christoph Schuhmann, Ludwig Schmidt, and Jenia Jitsev. Reproducible scaling laws for contrastive language-image learning. In *CVPR*, 2023. 1, 2, 4

[15] Chunyuan Deng, Yilun Zhao, Xiangru Tang, Mark Gerstein, and Arman Cohan. Investigating data contamination in modern benchmarks for large language models. In *NAACL*, 2024. 1

[16] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *CVPR*, 2009. 1, 2, 4

[17] Jesse Dodge, Maarten Sap, Ana Marasović, William Agnew, Gabriel Ilharco, Dirk Groeneveld, Margaret Mitchell, and Matt Gardner. Documenting large webtext corpora: A case study on the colossal clean crawled corpus. In *EMNLP*, 2021. 1

[18] Matthijs Douze, Alexandr Guzhva, Chengqi Deng, Jeff Johnson, Gergely Szilvasy, Pierre-Emmanuel Mazaré, Maria Lomeli, Lucas Hosseini, and Hervé Jégou. The Faiss library, 2025. 3

[19] Yanai Elazar, Akshita Bhagia, Ian Helgi Magnusson, Abhilasha Ravichander, Dustin Schwenk, Alane Suhr, Evan Pete Walsh, Dirk Groeneveld, Luca Soldaini, Sameer Singh, et al. What's in my big data? In *ICLR*, 2024. 1

[20] Samir Yitzhak Gadre, Gabriel Ilharco, Alex Fang, Jonathan Hayase, Georgios Smyrnis, Thao Nguyen, Ryan Marten, Mitchell Wortsman, Dhruba Ghosh, Jieyu Zhang, et al. DataComp: In search of the next generation of multimodal datasets. In *NeurIPS*, 2024. 1

[21] Noa Garcia, Yusuke Hirota, Yankun Wu, and Yuta Nakashima. Uncurated image-text datasets: Shedding light on demographic bias. In *CVPR*, 2023. 1

[22] R Girshick. Fast R-CNN. In *ICCV*, 2015. 3

[23] Shahriar Golchin and Mihai Surdeanu. Time travel in LLMs: Tracing data contamination in large language models. In *ICLR*, 2024. 1

[24] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 4

[25] Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. Mistral 7B, 2023. 1

[26] Justin Johnson, Bharath Hariharan, Laurens Van Der Maaten, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. CLEVR: A diagnostic dataset for compositional language and elementary visual reasoning. In *CVPR*, 2017. 2

[27] Andrej Karpathy and Li Fei-Fei. Deep visual-semantic alignments for generating image descriptions. In *CVPR*, 2015. 4

[28] Douwe Kiela, Hamed Firooz, Aravind Mohan, Vedanuj Goswami, Amanpreet Singh, Pratik Ringshia, and Davide Testuggine. The Hateful Memes Challenge: Detecting hate speech in multimodal memes. In *NeurIPS*, 2020. 2

[29] Alina Kuznetsova, Hassan Rom, Neil Alldrin, Jasper Uijlings, Ivan Krasin, Jordi Pont-Tuset, Shahab Kamali, Stefan Popov, Matteo Malloci, Alexander Kolesnikov, et al.

The Open Images Dataset V4: Unified image classification, object detection, and visual relationship detection at scale. *IJCV*, 2020. 4

[30] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proc. IEEE*, 1998. 2

[31] Chunyuan Li, Cliff Wong, Sheng Zhang, Naoto Usuyama, Haotian Liu, Jianwei Yang, Tristan Naumann, Hoifung Poon, and Jianfeng Gao. LLaVA-Med: Training a large language-and-vision assistant for biomedicine in one day. In *NeurIPS*, 2024. 1

[32] 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 *ECCV*, 2014. 1, 2, 3

[33] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin Transformer: Hierarchical Vision Transformer using shifted windows. In *ICCV*, 2021. 3

[34] Inbal Magar and Roy Schwartz. Data contamination: From memorization to exploitation. In *ACL*, 2022. 1

[35] Dhruv Mahajan, Ross Girshick, Vignesh Ramanathan, Kaiming He, Manohar Paluri, Yixuan Li, Ashwin Bharambe, and Laurens Van Der Maaten. Exploring the limits of weakly supervised pretraining. In *ECCV*, 2018. 2

[36] Nicole Meister, Dora Zhao, Angelina Wang, Vikram V Ramaswamy, Ruth Fong, and Olga Russakovsky. Gender artifacts in visual datasets. In *ICCV*, 2023. 1

[37] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. DINOv2: Learning robust visual features without supervision. In *ICLR*, 2024. 4

[38] Yonatan Oren, Nicole Meister, Niladri S Chatterji, Faisal Ladhak, and Tatsunori Hashimoto. Proving test set contamination in black-box language models. In *ICLR*, 2024. 1

[39] Aleksandra Piktus, Christopher Akiki, Paulo Villegas, Hugo Laurençon, Gérard Dupont, Sasha Luccioni, Yacine Jernite, and Anna Rogers. The ROOTS search tool: Data transparency for LLMs. In *ACL*, 2023. 1

[40] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *ICML*, 2021. 1, 2, 3, 4, 7

[41] Mathieu Ravaut, Bosheng Ding, Fangkai Jiao, Hailin Chen, Xingxuan Li, Ruochen Zhao, Chengwei Qin, Caiming Xiong, and Shafiq Joty. How much are LLMs contaminated? a comprehensive survey and the llmsanitize library, 2025. 1

[42] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. *PAMI*, 2016. 3

[43] Manley Roberts, Himanshu Thakur, Christine Herlihy, Colin White, and Samuel Dooley. To the cutoff... and beyond? a longitudinal perspective on LLM data contamination. In *ICLR*, 2023. 1

[44] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, 2022. 4

[45] Oscar Sainz, Jon Campos, Iker García-Ferrero, Julen Etxaniz, Oier Lopez de Lacalle, and Eneko Agirre. NLP evaluation in trouble: On the need to measure LLM data contamination for each benchmark. In *EMNLP Findings*, 2023. 1

[46] Christoph Schuhmann, Robert Kaczmarczyk, Aran Komatsuzaki, Aarush Katta, Richard Vencu, Romain Beaumont, Jenia Jitsev, Theo Coombes, and Clayton Mullis. LAION-400M: Open dataset of CLIP-filtered 400 million image-text pairs. In *NeurIPSW*, 2021. 1, 2, 4

[47] Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, et al. LAION-5B: An open large-scale dataset for training next generation image-text models. In *NeurIPS*, 2022. 1

[48] Piyush Sharma, Nan Ding, Sebastian Goodman, and Radu Soricut. Conceptual Captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In *ACL*, 2018. 4

[49] Weijia Shi, Anirudh Ajith, Mengzhou Xia, Yangsibo Huang, Daogao Liu, Terra Blevins, Danqi Chen, and Luke Zettlemoyer. Detecting pretraining data from large language models. In *ICLR*, 2024. 1

[50] Oleksii Sidorov, Ronghang Hu, Marcus Rohrbach, and Amanpreet Singh. TextCaps: a dataset for image captioning with reading comprehension. In *ECCV*, 2020. 4

[51] Weijie Su, Xizhou Zhu, Yue Cao, Bin Li, Lewei Lu, Furu Wei, and Jifeng Dai. VL-BERT: Pre-training of generic visual-linguistic representations. In *ICLR*, 2020. 4

[52] Giorgos Tolias, Ronan Sicre, and Hervé Jégou. Particular object retrieval with integral max-pooling of CNN activations. In *ICLR*, 2016. 2

[53] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models, 2023. 1

[54] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and tell: A neural image caption generator. In *CVPR*, 2015. 3

[55] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The Caltech-UCSD Birds-200-2011 Dataset. Technical report, California Institute of Technology, 2011. 2

[56] Chien-Yao Wang, I-Hau Yeh, and Hong-Yuan Mark Liao. YOLOv9: Learning what you want to learn using programmable gradient information. In *ECCV*, 2025. 3

[57] Peter Young, Alice Lai, Micah Hodosh, and Julia Hockenmaier. From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions. *TACL*, 2014. 4

[58] Christoph Zauner. Implementation and benchmarking of perceptual image hash functions. Master's thesis, Upper Austria University of Applied Sciences, 2010. 2

[59] Xiaohua Zhai, Xiao Wang, Basil Mustafa, Andreas Steiner, Daniel Keysers, Alexander Kolesnikov, and Lucas Beyer. LiT: Zero-shot transfer with locked-image text tuning. In *CVPR*, 2022. 2

[60] Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 million image database for scene recognition. *PAMI*, 2017. 2