A. Appendix

A.1. Method details

A.1.1 Semantic parser

To enable a rich complexity and semantic filtering, we built a fast custom semantic parser that converts a given textual caption to a semantic graph similar to the one in Visual Genome [36]. In particular, we extract objects, their parts, their attributes, and the actions that they are involved in (see Figure 3 for example). The parser is built on top of the English language dependency parser from Spacy [26] combined with multiple rules to infer common object relations. The aim of the parser is high speed with high precision of common object relations such as 'has_attribute' and 'has_part' and basic 'action' support. Below, we describe the structured relations that we extract from natural language text.

We support the following semantic relations:

Object (_obj). We extract objects that are supposedly presented in an image. We consider nouns that are not attributes of another noun (not part of a noun phrase). *E.g.* in *birthday cake* and *baby stroller*, the nouns *cake* and *stroller* are parsed as objects, and the nouns *birthday* and *baby* are considered attributes. We do not consider proper nouns.

Attribute (has_attr). Denotes attributes that characterize an object or another attribute. For example, *dark green*, would result in a fact *green - has_attr - dark*, and *yellow candles* results in *candles - has_attr - yellow*.

Part (has_part). Characterizes a visual part of an object. *E.g. cake with 21 yellow candles* would result in a part fact *cake - has_part - candles*.

Action (_act). Verbs that do not entail attributes or parts (*e.g.* forms of *be*, *looks*, *seems*, and *have* are excluded) are considered actions. For actions, we also parse the subject and object arguments.

Subject of an action (act_has_subj, is_act_subj). We use the act_has_subj and is_act_subj relation to represent arguments (nouns) that are the subject of an action. *E.g.* for the text *a person is eating an apple*, we add the object-centric and corresponding action-centric symmetric facts: *person - is_subj_act - eating* and *eating act_has_subj person*.

Object of an action (act_has_obj, is_act_obj). We also include the relations that specify the object arguments of an action. *E.g.* for the text *a person is eating an apple*, we add the object-centric and corresponding action-centric symmetric facts: *apple - is_obj_act - eating* and *eating act_has_obj apple*.

We recognize the following limitations of ours approach:

Semantic attributes. In this work, we focus on object-centric visual and action characteristics and we do not process spatial relations (X next to Y) or additional action arguments (read a book *in* the library). Spatial relations and additional arguments of verbs usually involve more complex semantic reasoning and require more robust approaches and task-specific models such as one trained on Semantic Role Labeling which are usually compute-heavy. We leave these for future work.

Dependency parser errors. In the current version of the parser, we also parse potential attributes as actions, which are not likely to be always visual. *E.g.* In the phrase "running person", running is an action and an attribute, and we parse them as such. However, sometimes the underlying parser would also parse attributes in phrases such as "striped mug" as verbs, where we process the attribute "striped" as both an attribute and an action (without arguments).

A.1.2 Concept distillation

The teacher model is built by training linear classifiers - which predict objects and attributes - on top of a frozen SWAG [71] backbone. SWAG is trained in a weakly-supervised manner by predicting hashtags from Instagram images. We use the publicly available weights, and adopt a training procedure that is similar to the one from SWAG for learning the linear classifiers. The procedure for training the object classifier is as follows. First, we parse the captions to extract nouns. Next, we canonicalize the nouns via WordNet [52] synsets and remove ones which occur less than 250 times in the dataset. The resulting vocabulary contains $\sim$10K unique synsets. Finally, we optimize the linear layer's weights through a cross-entropy loss. Each entry in the target distribution of the cross-entropy is either $1/K$ or 0 depending on whether the corresponding synset is present or not, where K is the number of synsets for that image. We apply inverse square-root resampling of images to upsample the tail classes following [71]. The target length of the dataset is set to 50 million samples during resampling . We train the linear layer using SGD with momentum 0.9 and weight decay 1e-4. The learning rate is set following the linear scaling rule: lr=$0.001 \cdot \frac{bs}{256}$. To speedup training, we use 64 GPUs with batch size of 256 per GPU. The attribute classifiers are build in a similar way, but the WordNet adjective synsets require additional filtering to remove non-visual attributes, *e.g.*, *claustrophobic*, *experienced*. Following [61], we select the attributes based on their *sharedness* and *visualness*. We rank the attributes based on the aforementioned scores, and keep $\sim$1200 attributes.

Table A.1. DiHT architecture hyperparameters.

Model	Dim	Vision			Language		
		layers	width	heads	layers	width	heads
B/32	512	12	768	12	12	512	8
B/16	512	12	768	12	12	512	8
L/14	768	24	1024	16	12	768	12

Table A.2. DiHT common hyperparameters.

Shared			
Learning rate (LR)	1e-3		
Warm-up	1%		
Vocabulary size	49408		
Temperature (init, max)	$(\frac{1}{0.07}, 100.0)$		
Adam (β_1, β_2)	(0.9, 0.98)		
Adam ϵ	1e-6		
High resolution LR	1e-4		
Dataset specific	LAION		PMD
CD learning rate scale	10.0		1.0
CD weight decay scale	0.01		1.0
HN-NCE α	1.0		0.999
HN-NCE β	0.25		0.5
	LAION		PMD
Model specific	L/14	B/16,B/32	B/16,B/32
Batch size	98304	49152	32768
Weight decay	0.2	0.1	0.1

A.2. Training details

For our model architecture, we closely follow CLIP by Radford *et al.* [62]. We utilize Vision Transformers (ViT) [17] for images and Text Transformers [75] for captions. We experiment with 3 different architectures, denoted as B/32, B/16, and L/14, where 32, 16, and 14 denote the input image patch size. Other architecture scaling parameters are in Table A.1. For distillation and fine-tuning experiments, we utilize the public SWAG-ViT models [71], pre-trained with weak supervision from hashtags.

We use the Adam [33] optimizer with a decoupled weight decay [48] and a cosine learning rate schedule [47]. Input image size is 224×224 pixels, for pre-training runs. All hyperparameters are presented in Table A.2. They are selected by training on a small scale setup, and reused for other experiments. For objects and attributes classifiers in concept distillation (CD), we found that scaling the learning rate by 10.0 and weight decay by 0.01 gave better results.

We pre-train the models on 4B, 8B, 16B, or 32B processed samples, depending on the experiment. For L/14 we train at a higher 336px resolution for additional 400M samples, denoting this models as L/14@336. We trained L/14 for 6 days on 512 A100 GPUs with 16B processed samples for a total of 7.4×10^4 GPU hours.

To accelerate training and save memory, we use mixed-precision training [51]. For L/14 we use grad checkpointing [8] and BFLOAT16 [14,29] format, all the other models are trained using FP16 [51] format. Contrastive loss is computed on the local subset of the pairwise similarities [62].

A.3. Evaluation details

We evaluate our models on a zero-shot benchmark of 24 datasets: (i) **17 image classification**: Birdsnap [3], CIFAR10 [37], CIFAR100 [37], Caltech101 [19], Country211 [62], DTD [13], Flowers102 [54], Food101 [4], ImageNet1K [65], OxfordPets [57], STL10 [15], SUN397 [79], StanfordCars [35], UCF101 [72], HatefulMemes [32], PascalVOC2007 [18], OpenImages [39]; (ii) **5 cross-modal retrieval** (text-to-image T2I, image-to-text I2T): COCO [45], Flickr [59], LN-COCO [60], LN-Flickr [60], Winoground [73]; (iii) **2 visual question answering**: SNLI-VE [80], VQAv2 [21]. Note that, cross-modal retrieval datasets have 2 tasks (T2I and I2T), so in total we evaluate across 29 tasks.

We follow zero-shot CLIP benchmark[7] implementation for most of the datasets, and implement the ones that are missing. For most image classification tasks we compute Accuracy@1, except HatefulMemes where we compute AUROC because it is binary classification, OpenImages where we compute FlatHit@1 following [77], and PascalVOC2007 where we compute mean average precision (mAP) because it is multi-label classification. We use the same prompt ensembling method as CLIP [62] to improve zero-shot image classification. For cross-modal retrieval (T2I and I2T), we compute Recall@1. For COCO and Flickr we apply a simple prompt pretext "a photo of {caption}", for LN-COCO, LN-Flickr, and Winoground no prompt is applied. We cast visual question answering (VQA) as binary prediction task and compute AP on the cosine similarity between an image and a text (a hypothesis or a question). For SNLI-VE, we take a subset which has agreement among annotators, we use "entailement" and "contradiction" as binary classes, and drop the "neutral" class. For VQAv2, we take the subset with yes/no questions. No prompt is applied for SNLI-VE and VQAv2.

A.4. Additional ablations

Effect of dataset filtering. In Figure A.1 we observe that gains from our proposed complexity, action, and text-spotting (CAT) dataset filtering hold as we train for longer training schedules. We ran small scale experiments with several complexity filters (see Table A.3) and we found that CAT with minimum complexity C1 performed the best.

Effect of top-k predicted objects and attributes. In Table A.4, we show that our concept distillation approach is quite robust to the choice of the number of predicted objects and attributes. For $k = 10$ strong accuracy is achieved with a small increase in dataset memory.

[7]github.com/LAION-AI/CLIP_benchmark

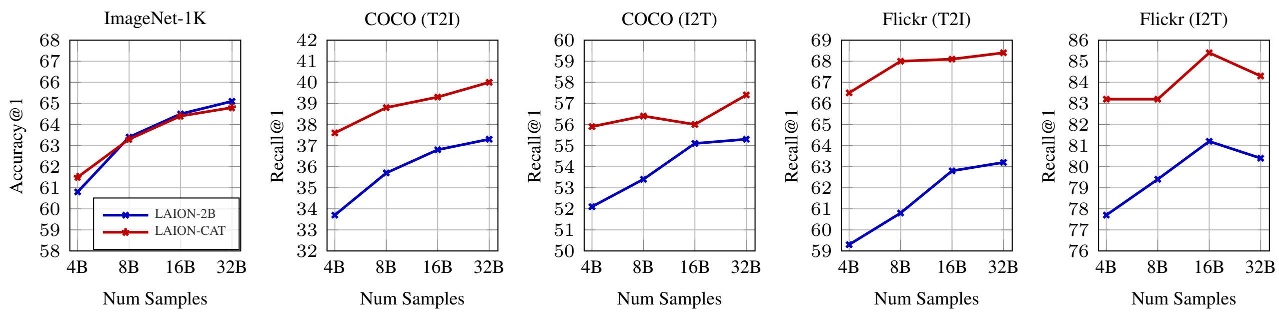

Figure A.1. Evaluating effect of using our LAION-CAT subset filtered on complexity (C), actions (A), and text spotting (T). Evaluation performed on ViT-B/32 architecture trained for a varying number of processed samples.

Table A.3. Number of examples after filtering with different filters.

Filter						# examples	% of full
[69]	C0	C1	C2	A	T		
						2,121,505,329	100.00
✓						1,983,345,180	93.49
✓	✓					1,891,725,045	89.17
✓		✓				1,709,522,548	80.58
✓			✓			1,143,660,096	53.91
✓				✓		691,535,901	32.60
✓		✓		✓		642,162,957	30.27
✓			✓	✓		487,493,190	22.98
✓		✓		✓	✓	438,358,791	20.66

Table A.4. Evaluating effect of using different number of top-k predicted objects and attributes. Evaluation on ViT-B/16 model architecture trained for 8B processed samples on LAION-CAT. Memory denotes storage needed to store predicted concepts.

top-k	Memory	IN	COCO		Flickr	
			T2I	I2T	T2I	I2T
5	16.3GB	71.4	42.9	59.4	72.2	86.5
10	32.6GB	71.9	42.9	60.3	73.3	87.0
25	81.6GB	71.4	43.1	60.0	72.9	87.9

Table A.5. Evaluating effect of different hyperparameters α and β for the HN-NCE loss. Evaluation on ViT-B/16 model architecture trained for 16B processed samples on LAION-CAT.

α	β	IN	COCO		Flickr	
			T2I	I2T	T2I	I2T
1	0	68.7	42.8	60.5	72.8	87.6
1	0.25	69.2	42.9	61.2	72.6	87.8
1	0.5	66.5	40.3	59.7	71.4	84.9
0.999	0.25	69.0	42.6	60.9	72.3	87.9
0.9	0.25	68.6	42.1	59.2	71.2	85.5

Table A.6. Evaluating linear probing with the complete training set for ImageNet1K on the ViT-L/14 architecture.

Model	Optimizer	ImageNet-1K Accuracy (%)
CLIP-L/14 @ 224px	SGD	83.60
DiHT-L/14 @ 224px	SGD	85.40
DiHT-L/14 @ 224px	PGD	**85.41**
CLIP-L/14 @ 336px	SGD	85.40
DiHT-L/14 @ 336px	SGD	85.87
DiHT-L/14 @ 336px	PGD	**85.89**

Effect of α and β on HN-NCE. From intuition, one can see that the term α controls the mass of the positive alignment term in the loss function, and the term β controls the difficulty of the negatives. The need for the term α can be attributed as follows. If there are false negatives within the dataset, dampening the positive alignment term can prevent the model from becoming overly discriminative with the true and false positive pairs. Hence, we would like to reduce α as the likelihood of having false positives increases (*e.g.*, smaller datasets, less noisy training). The need for β is straightforward: higher β pushes the weighing function to be "sharper", with more mass on the hardest negatives. Table A.5 shows the effect of different values of α and β on LAION-CAT.

Additional results on few-shot probing. We examine the performance of our models on linear probing with the full training set for ImageNet1K [65]. We compare the performance of DiHT-L/14 and CLIP-L/14 [62] architectures for both the 224px and 336px input sizes in Table A.6. We observe that the PGD approach with the DiHT model outperforms prior work, and also find that there is no notable difference in performance between SGD-trained and PGD-trained models, as there is no need for regularization when training with the full dataset. We reproduce the reported numbers for CLIP [62] and train our models with a learning rate of 24, no weight decay, and batch size of 96,000 for 160 epochs.

Table A.7. Zero-shot state-of-the-art dual-encoder models comparison. We evaluate CLIP [62] and OpenCLIP [27] using our codebase.

Method	Birdsnap	CIFAR10	CIFAR100	Caltech101	Country211	DTD	Flowers102	Food101	ImageNet1K	OxfordPets	STL10	SUN397	StanfordCars	UCF101	HatefulMemes	PascalVOC	OpenImages	COCO T2I	COCO I2T	Flickr T2I	Flickr I2T	LN-COCO T2I	LN-COCO I2T	LN-Flickr T2I	LN-Flickr I2T	Winoground T2I	Winoground I2T	SNLI-VE	VQAv2
ViT-B/32 @ 224																													
CLIP	40.3	89.8	65.1	83.9	17.2	43.8	66.6	83.9	63.4	87.4	97.2	62.3	59.7	64.2	58.1	84.2	27.8	31.4	49.0	59.5	79.9	16.8	24.6	30.2	38.1	28.1	27.4	77.6	57.3
OpenCLIP	50.5	93.6	75.8	86.4	16.7	56.1	71.7	82.7	66.6	90.6	96.6	68.5	86.0	66.1	53.4	85.4	34.6	39.0	56.7	65.7	81.7	29.5	35.1	44.0	51.4	32.0	30.2	78.6	59.3
DiHT	46.5	92.0	73.6	80.4	16.3	55.3	69.8	84.1	68.0	91.7	97.2	66.5	79.6	68.3	53.5	78.9	32.4	40.6	59.3	68.6	84.4	29.8	35.7	46.1	54.0	30.9	33.0	79.1	59.9
ViT-B/16 @ 224																													
CLIP	43.2	90.8	68.3	84.7	22.8	44.9	71.2	88.7	68.4	89.1	98.3	64.4	64.7	69.5	59.3	85.3	29.3	33.7	51.3	63.3	81.9	18.7	25.2	31.3	37.4	31.0	30.2	77.9	57.7
OpenCLIP	52.1	91.7	71.4	86.2	18.1	50.8	69.3	86.1	67.1	89.4	97.0	69.6	83.8	67.7	55.7	84.2	35.2	37.8	55.4	65.2	84.1	26.1	33.1	43.5	46.9	30.5	30.2	78.4	59.3
DiHT	54.5	92.7	77.5	81.2	19.1	59.4	70.5	89.1	72.2	92.7	98.2	68.4	86.0	70.3	56.2	79.5	34.6	43.3	60.3	72.9	89.8	32.4	38.2	52.9	57.7	32.0	33.4	80.8	60.3
ViT-L/14 @ 224																													
CLIP	52.5	95.6	78.2	86.7	31.9	55.5	79.1	93.1	75.6	93.5	99.4	67.6	77.8	77.0	60.4	85.5	30.6	36.5	54.9	66.1	84.5	20.8	28.6	36.2	44.2	31.9	32.0	78.2	58.4
OpenCLIP	62.9	96.6	83.4	88.0	26.3	62.9	75.5	91.0	75.2	93.2	98.9	74.3	92.6	75.2	55.1	87.5	38.0	46.2	64.3	75.4	90.4	34.6	39.9	50.9	57.7	33.4	36.4	80.8	60.0
DiHT	60.4	91.7	81.3	81.6	26.0	60.3	77.6	92.7	77.0	93.8	98.0	70.2	91.1	77.9	56.5	79.3	35.0	48.0	65.1	76.7	92.0	35.6	40.7	52.7	60.3	31.8	33.4	81.3	61.0
ViT-L/14 @ 336																													
CLIP	53.7	95.0	77.0	87.2	34.4	56.0	78.6	93.8	76.6	93.8	99.5	68.7	79.2	77.6	61.6	86.2	31.8	37.7	57.1	68.6	86.6	20.2	28.6	38.1	45.7	32.3	21.4	78.7	58.5
DiHT	62.0	92.2	81.2	82.4	27.8	61.1	77.0	92.9	77.9	94.0	98.2	71.2	91.5	77.7	56.3	81.0	36.5	49.3	65.3	78.2	91.1	36.7	41.2	54.5	61.6	35.0	38.5	81.7	61.4

Additional results on zero-shot benchmark. We report performance of CLIP [62], OpenCLIP [27], and DiHT on all 29 zero-shot tasks in Table A.7.

A.5. Contrastive Alignment with Hard Negatives

Convergence guarantees

Proposition 1. *Let* $\mathcal{L}^\star(\phi_i, \phi_t) = \sup_{q \in \Pi} \mathcal{L}_(\phi_i, \phi_t, q)$. *Then for any measurable* $\phi_i, \phi_t : \mathcal{X} \to \mathbb{S}^{d-1}$ *and* $\tau = \mathcal{O}(1)$ *we observe the convergence* $\mathcal{L}_(\phi_i, \phi_t, q) \to \mathcal{L}^\star(\phi_i, \phi_t)$ *as* $\beta \to \infty$.

Proof. Follows from Proposition 6 of [63] with the loss function $\mathcal{L}(\phi_i, \phi_t, q_\beta)$ defined as follows for any β.

$$\mathcal{L}(\phi_i, \phi_t, q_\beta) =$$
$$\log\left[\frac{e^{\phi_i(x)^\top \phi_t(x)/\tau}}{e^{\phi_i(x)^\top \phi_t(x)/\tau} + Q \cdot \mathbb{E}_{y \sim q_\beta}\left[e^{\phi_i(x)^\top \phi_t(y)/\tau}\right]}\right]$$
$$+ \log\left[\frac{e^{\phi_i(x)^\top \phi_t(x)/\tau}}{e^{\phi_i(x)^\top \phi_t(x)/\tau} + Q \cdot \mathbb{E}_{y \sim q_\beta}\left[e^{\phi_i(x)^\top \phi_t(y)/\tau}\right]}\right].$$

$\square$