A. Appendix

A.1. Compare with GPT-4V Preview

Since the GPT-4V(ision) Preview [86] is also able to generate object labels for images, we compare our method with it for the recognition task. The API parameters for the GPT-4V Preview [86] are: input image size is 256^2, temperature is zero for deterministic predictions, and detail is low with sampling 65 output tokens. The model version from API is `gpt-4-1106-vision-preview`. We prompt it to generate ten main object labels as its top-10 predictions with the following instruction:

the instruction for OpenAI GPT-4-vision-preview API[5]
Describe every detail in the image by listing ten main object labels. The answer should only contain the object labels separated by a comma, for example, "car, airplane, dog".

Due to the API request limit, we are able to evaluate it on a subset of the COCO validation split, which contains 4359 out of 5000 images in total. We compare various methods in Table A.1 with top-10 predictions, showing that our method performs better than the GPT-4V Preview [86] across all metrics, and the GPT-4V Preview has the second-highest R. The PR-curves are illustrated in Figure A.1, indicating that our method has a better P/R trade-off. Since GPT-4V Preview consistently generates ten labels for each image, its P is also low compared to Flamingo$_{\text{open}}$ and InstructBLIP.

		COCO		
method	prompt	R	P	F_1
CLIP [93]	-	0.525	0.562	0.540
Flamingo$_{\text{open}}$ [3] w/ MPT [111]	list	0.556	0.794	0.647
InstructBLIP [22]	list	0.613	**0.897**	0.725
GPT-4V Preview [86]	instruct	0.625	0.601	0.610
Ours	-	**0.765**	0.756	**0.758**

Table A.1. **Comparison with top-10 predictions** on COCO validation subset.

Cross-Validation. As we mentioned in Section 3.3, the reference labels extracted from the raw captions are imperfect and incomplete. To verify that our method generalizes well to predict plausible labels, we conduct a cross-validation on the COCO validation subset, treating the GPT-4V Preview's predictions as reference labels to evaluate others. Table A.2 demonstrates that our method consistently matches the performance across all metrics as presented in Table 1, in which our method ranks first in R and F_1. Again, the lower P for our method is due to the fact that our model predicts the required number of labels, while others with a higher P presumably predict less than ten labels. Regarding R, LLaVA$_{1.0}$ [69] ranks second in performance.

[5] platform.openai.com/docs/guides/vision.

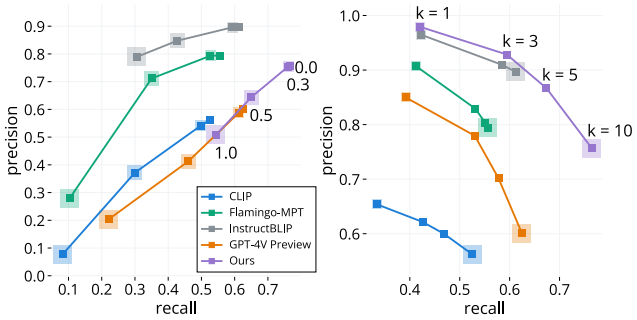


Figure A.1. **Precision-recall (PR) curves** on COCO validation subset. The same settings as in Figure 5.

		COCO		
method	prompt	R	P	F_1
CLIP [93]	-	0.467	0.509	0.485
CaSED [19]	-	0.535	0.562	0.546
Flamingo$_{\text{open}}$ [3] w/ MPT [111]	list	0.517	**0.760**	0.609
LLaVA$_{1.0}$ [69]	caption	0.593	0.599	0.595
LLaVA$_{1.5}$ [68]	caption	0.576	0.572	0.573
BLIP-2 [65]	caption	0.498	0.736	0.590
InstructBLIP [22]	list	0.505	0.731	0.594
GPT-4V Preview [86]	instruct	1.000	1.000	1.000
Ours	-	**0.632**	0.651	**0.641**
Ours w/ top-100	-	0.823	0.473	0.600

Table A.2. **Comparison with top-10 predictions** on COCO validation subset, viewing GPT-4V Preview's predictions as reference labels. Gray row shows our top-100 predictions.

A.2. Ranking Predictions

We ablate ranking strategies for the predictions produced by our model. Given an image, our model generates K labels $\mathcal{L} = \{L_1, \ldots, L_K\}$. Each label L_k has $T_k + 1$ tokens, including the special token [SEP] for the delimiter.

Ranking by CLIP Score. The first strategy is to rank the predictions by the CLIP score:

$$\text{clip}(L_k) = f_{\text{CLIP}}(\text{image}, \text{label } L_k), \tag{A.1}$$

where f_{CLIP} is the CLIP model [93] with the image encoder of ViT-L/14 and the language encoder. The CLIP score is based on cosine distance in the embedding space.

Ranking by Probability. The second strategy is to rank the predictions by their probabilities in Eq. 6:

$$\text{prob}(L_k) = \prod_{t=1}^{T_k+1} P(\mathbf{w}_t^k | \mathbf{w}_{<t}^k, \mathbf{X}), \tag{A.2}$$

in which the probability of each label is the product of the individual probabilities of its tokens, including the delimiter token [SEP]. If greedy and beam search sample a particular label multiple times, we sum up the probabilities as its final probability.

Ranking by Perplexity. The third one is to rank the predictions by their perplexities. The perplexity is computed with the fixed length $T_k + 1$ for each label:

$$\text{ppl}(L_k) = \exp\left[-\frac{1}{T_k+1}\sum_{t=1}^{T_k+1}\log P(\mathbf{w}_t^k|\mathbf{w}_{<t}^k, \mathbf{X})\right]. \quad \text{(A.3)}$$

If the greedy and beam search sample a particular label multiple times, we use its minimum perplexity to ensure optimal selection and accuracy.

Ranking by Cross-Modal Similarity Score. The last one is to rank predictions by their cross-modal similarity scores, computed with the image and label token embeddings:

$$\text{sim}(L_k) = \frac{1}{T_k}\sum_{t=1}^{T_k} d(\mathbf{w}_t^k, \mathbf{X}_\text{v}), \quad \text{(A.4)}$$

where d is the euclidean distance averaged over all the image token embeddings for each label token embedding $\mathbf{w}_t^k$:

$$d(\mathbf{w}_t^k, \mathbf{X}_\text{v}) = \frac{1}{M}\sum_{i=1}^{M}\sqrt{2 - 2\cdot\frac{\mathbf{w}_t^k\cdot\mathbf{x}_i^\text{v}}{\|\mathbf{w}_t^k\|_2\cdot\|\mathbf{x}_i^\text{v}\|_2}}, \quad \text{(A.5)}$$

where M is the number of image tokens. This similarity is also called compatibility score to measure the compatibility between image and label embeddings, which motivates us to select the predictions that are compatible with the corresponding images. In other words, the closer the label token embeddings are to the image token embeddings, the more likely the label is the correct prediction.

Results. Table A.3 compares the above four ranking strategies using top-5 predictions across different sampling methods for our 1.78B model trained on G3M. The greedy and 3-way beam search samples 64 tokens for each image. Since one-shot sampling yields ordered predictions, we sample 10 labels per image and utilize ranking strategies to select the final top-5 predictions.

The overall best ranking strategy is using probability for greedy search and one-shot sampling, and using CLIP score for beam search. For R, one-shot sampling with probability ranks first on CC3M and COCO, and the greedy search with probability leads on OpenImages. The greedy search with probability has a slightly higher P than one-shot sampling with probability, but the latter has a better overall F_1.

For greedy search, the compatibility score has the same performance as the perplexity. For one-shot sampling, the compatibility score is better than the perplexity. Without a ranking strategy, one-shot sampling matches the performance of probability-based ranking, showing its effectiveness in using top-k initial tokens to decide the final top-k predictions.

No ranking strategy outperforms the CLIP score for both greedy and beam search, yet we apply CLIP score to other models like Flamingo, BLIP-2, InstructBLIP, and LLaVA.

	greedy			beam			one-shot		
ranking	R	P	F_1	R	P	F_1	R	P	F_1
CC3M									
-	0.661	**0.604**	0.624	0.641	0.590	0.608	0.673	0.598	0.627
clip	0.646	0.604	0.617	0.630	0.594	0.605	0.643	0.588	0.608
prob	0.659	0.602	0.622	-	-	-	**0.673**	0.598	**0.627**
ppl	0.614	0.563	0.581	-	-	-	0.509	0.466	0.484
sim	0.611	0.564	0.581	0.598	0.557	0.571	0.594	0.531	0.556
COCO									
-	0.606	**0.802**	0.687	0.585	0.772	0.663	0.618	0.799	0.695
clip	0.590	0.792	0.673	0.573	0.772	0.654	0.592	0.773	0.668
prob	0.603	0.796	0.683	-	-	-	**0.619**	0.800	**0.695**
ppl	0.578	0.748	0.649	-	-	-	0.528	0.640	0.577
sim	0.576	0.747	0.647	0.552	0.724	0.623	0.576	0.717	0.637
OpenImages									
-	0.549	0.599	0.565	0.530	0.577	0.546	0.560	0.595	0.570
clip	0.540	**0.598**	0.560	0.525	0.580	0.544	0.543	0.591	0.559
prob	**0.580**	0.576	0.569	-	-	-	0.562	0.597	**0.572**
ppl	0.577	0.571	0.565	-	-	-	0.495	0.505	0.496
sim	0.575	0.571	0.564	0.509	0.553	0.524	0.527	0.547	0.532

Table A.3. **Comparison of different ranking strategies for various sampling methods** with top-5 predictions. In the case of "-", no ranking strategy is used, and one-shot sampling directly outputs the top-5 labels.

	CC3M			COCO			OpenImages		
ranking	R	P	F_1	R	P	F_1	R	P	F_1
-	0.545	0.568	0.549	0.548	0.794	0.643	0.526	0.655	0.576
clip	0.551	0.574	0.555	0.552	0.801	0.648	0.527	0.657	0.577

Table A.4. **Comparison of different ranking strategies** with top-5 predictions for $\text{Flamingo}_{\text{open}}$ + MPT.

For BLIP-2, InstructBLIP, and LLaVA, whose outputs are sentences, the CLIP score is the only choice for ranking. But for Flamingo, since it has a same format as ours, we can test its performance without ranking strategy. Because it saturates at top-10, we only report its top-5 comparison. The results are shown in Table A.4, showing that the CLIP score is the optimal ranking strategy for those models.

A.3. Additional Results

In this section, we present additional results, mainly with top-10 predictions, for ablation studies.

Ablation on Truncating the Decoder. We compare the results of different truncating sizes of the language decoder with top-10 predictions in Table A.5. There is a small performance drop, $0.745 \rightarrow 0.738$ in R on CC3M, with truncating the decoder from 3B to 1.78B, while the performances on COCO and OpenImages remain the same.

Ablation on Sampling Methods. We compare sampling methods, i.e., greedy search, 3-way beam search, and one-shot sampling, with top-10 predictions in Table A.6. The results, consistent with those in Table 5, indicate that one-shot sampling surpasses greedy and beam search in R and

F_1 scores but falls short in P when considering top-10 predictions. The reason is that greedy and beam search produce ∼7 labels average per image in top-10 due to the repetition issue. Figure A.2 (right side) demonstrates saturation around $k = 7$, accounting for their higher P in top-10 predictions. This ablation study shows that greedy and beam search do not produce more diverse predictions with increasing number of tokens.

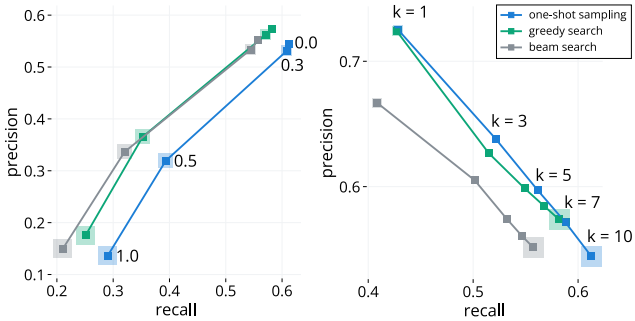


Figure A.2. **Precision-recall (PR) curves of different sampling methods** on OpenImages validation split with top-10 predictions. The same settings as in Figure 5.

Ablation on LLaMA Versions. Table A.7 compares the results of different LLaMA versions for the language decoder with top-10 predictions. The top-10 results are consistent with Table 7, showing LLaMA 2 is slightly better than LLaMA 1 on G3M, and comparable on G70M.

Ablation on Embedding Models in Evaluation Metric. The evaluation metric is based on embedding models to compute the similarity $\mathbf{S}_{ij}$ in Eq. 7. To verify the robustness of our method, we compare the results using CLIP ViT-L/14 [93] as the metric embedding model in Table A.8. Our results are from the 1.78B model trained on G70M, and the others are from the best settings in Table 1. Our method consistently outperforms others in R and F_1 scores, and is competitive in P.

Ablation on Training Epochs. We conduct an ablation study on training epochs for our 1.78B model on G3M. Table A.9 shows the results with top-10 predictions, indicating that training more epochs improves the performance.

Additional Main Results. Table A.10 shows the main results with top-5 predictions, consistent with those in Table 1. The performance drop on CC3M for models trained on G3M versus G70M stems from a data distribution shift.

A.4. Evaluation Metric

The recall in evaluation metric Eq. 8 essentially represents the top-k accuracy, which is for recognition tasks [99].

For an image, ground-truth (GT) labels are $\mathcal{G} = \{g_i\}_{i=1}^{M}$, ordered model predictions are $\mathcal{P} = \{p_j\}_{j=1}^{N}$. The standard recall is defined as $R_{ecall} = TP/(TP + FN)$.

# params	CC3M R	CC3M P	CC3M F_1	COCO R	COCO P	COCO F_1	OpenImages R	OpenImages P	OpenImages F_1
7.05B - 32	0.748	0.534	0.617	0.699	0.710	0.702	0.613	0.543	0.569
3.00B - 11	0.745	0.532	0.615	0.703	0.716	0.707	0.615	0.546	0.572
1.78B - 6	0.738	0.530	0.611	0.698	0.712	0.702	0.613	0.544	0.570
1.18B - 3	0.736	0.530	0.611	0.697	0.713	0.703	0.612	0.547	0.571
0.77B - 1	0.731	0.529	0.608	0.693	0.708	0.698	0.609	0.547	0.569

Table A.5. **Comparison of different language decoder sizes** with top-10 predictions. The same settings as in Table 3.

sampling	CC3M R	CC3M P	CC3M F_1	COCO R	COCO P	COCO F_1	OpenImages R	OpenImages P	OpenImages F_1
greedy	0.708	**0.568**	**0.621**	0.655	**0.755**	0.696	0.582	**0.574**	0.569
beam	0.681	0.557	0.604	0.623	0.725	0.665	0.557	0.552	0.546
one-shot	**0.738**	0.530	0.611	**0.698**	0.712	**0.702**	**0.613**	0.544	**0.570**

Table A.6. **Comparison of different sampling methods** with top-10 predictions. The greedy and beam search sample 128 tokens for each image without ranking strategies.

version	CC3M R	CC3M P	CC3M F_1	COCO R	COCO P	COCO F_1	OpenImages R	OpenImages P	OpenImages F_1
trained on G3M									
1	0.738	0.530	0.611	0.698	0.712	0.702	0.613	0.544	0.570
2	**0.740**	**0.531**	**0.612**	**0.700**	**0.714**	**0.705**	**0.614**	**0.547**	**0.571**
trained on G70M									
1	**0.722**	**0.512**	**0.593**	**0.765**	**0.757**	**0.758**	**0.663**	**0.564**	**0.603**
2	0.721	0.512	0.593	0.765	0.756	0.758	0.662	0.563	0.602

Table A.7. **Comparison of truncating different LLaMA versions** for the language decoder with top-10 predictions.

method	CC3M R	CC3M P	CC3M F_1	COCO R	COCO P	COCO F_1	OpenImages R	OpenImages P	OpenImages F_1
CLIP	0.799	0.746	0.771	0.774	0.783	0.778	0.762	0.725	0.742
Flamingo	0.842	**0.842**	0.841	0.835	0.922	0.875	0.838	0.863	0.849
BLIP-2	0.864	0.838	0.850	0.854	**0.961**	0.904	0.822	**0.864**	0.841
InstBLIP	0.883	0.827	0.853	0.892	0.887	0.889	0.878	0.842	**0.859**
Ours	**0.908**	0.825	**0.864**	**0.915**	0.911	**0.913**	**0.881**	0.838	0.858

Table A.8. **Comparison with top-**10 **predictions** using CLIP ViT-L/14 [93] as the embedding model in evaluation metric.

epoch	CC3M R	CC3M P	CC3M F_1	COCO R	COCO P	COCO F_1	OpenImages R	OpenImages P	OpenImages F_1
1	0.654	0.487	0.553	0.620	0.623	0.620	0.591	0.520	0.548
2	0.698	0.509	0.583	0.659	0.667	0.661	0.604	0.528	0.558
3	0.738	0.530	0.611	0.700	0.712	0.702	0.613	0.544	0.570

Table A.9. **Comparison of different training epochs** with top-10 predictions.

For recognition tasks, GT should either be TP (correctly identified) or FN (missed), i.e., $TP + FN = |\mathcal{G}| = M$, then

$$R_{ecall} = \frac{TP}{TP + FN} = \frac{TP}{|\mathcal{G}|} = \frac{TP}{M}. \tag{A.6}$$

For closed-set recognition, $TP = \sum_{i=1}^{M} \mathbb{I}(g_i \in \mathcal{P})$, where $g_i \in \mathcal{P}$ is a greedy matching – correct prediction is exactly the same as g_i with maximum semantic similarity, e.g., $g_i =$

method	models (vision + lang)	prompt	data scale	# params (B)	CC3M R	CC3M P	CC3M F_1	COCO R	COCO P	COCO F_1	OpenImages R	OpenImages P	OpenImages F_1
CLIP [93]	ViT L-14 + $\text{CLIP}_{\text{lang}}$	-	400M	0.43	0.515	0.481	0.493	0.468	0.590	0.523	0.460	0.485	0.467
CaSED [19]	ViT L-14 + Retrieval	-	12M	0.43	0.577	0.520	0.541	0.533	0.666	0.590	0.490	0.506	0.492
CLIP [93]	ViT L-14 + $\text{CLIP}_{\text{lang}}$	-	400M	0.43	0.400	0.388	0.390	0.385	0.489	0.427	0.349	0.366	0.354
CaSED [19]	ViT L-14 + Retrieval	-	403M	0.43	0.571	0.521	0.539	0.532	0.683	0.596	0.498	0.526	0.505
$\text{Flamingo}_{\text{open}}$ [3]	ViT L-14 + LLaMA 1 [112]	list	2.1B	8.34	0.542	0.541	0.535	0.541	0.726	0.616	0.524	0.622	0.561
$\text{Flamingo}_{\text{open}}$	ViT L-14 + LLaMA 1	caption	2.1B	8.34	0.539	0.523	0.525	0.547	0.712	0.614	0.533	0.608	0.561
$\text{Flamingo}_{\text{open}}$	ViT L-14 + MPT [111]	list	2.1B	8.13	0.551	0.574	0.555	0.552	0.801	0.648	0.527	0.657	0.577
$\text{Flamingo}_{\text{open}}$	ViT L-14 + MPT	caption	2.1B	8.13	0.532	0.537	0.528	0.551	0.762	0.635	0.544	0.655	0.588
$\text{LLaVA}_{1.0}$ [69]	ViT L-14 + LLaMA 2 [113]	list	753K	13.3	0.537	0.522	0.522	0.574	0.790	0.659	0.545	0.632	0.578
$\text{LLaVA}_{1.0}$	ViT L-14 + LLaMA 2	caption	753K	13.3	0.588	0.520	0.547	0.601	0.755	0.667	0.545	0.557	0.545
$\text{LLaVA}_{1.0}$	ViT L-14 + LLaMA 2	instruct	753K	13.3	0.566	0.507	0.531	0.600	0.746	0.662	0.567	0.589	0.571
$\text{LLaVA}_{1.5}$ [68]	ViT L-14 + Vicuna [16]	list	1.2M	13.4	0.535	0.523	0.521	0.581	0.800	0.666	0.545	0.618	0.573
$\text{LLaVA}_{1.5}$	ViT L-14 + Vicuna	caption	1.2M	13.4	0.581	0.510	0.543	0.600	0.751	0.664	0.551	0.560	0.555
$\text{LLaVA}_{1.5}$	ViT L-14 + Vicuna	instruct	1.2M	13.4	0.552	0.530	0.532	0.589	0.786	0.667	0.566	0.607	0.576
BLIP-2 [65]	ViT g-14 + Flant5xxl [17]	list	129M	12.2	0.541	0.558	0.541	0.482	0.842	0.606	0.466	0.626	0.526
BLIP-2	ViT g-14 + Flant5xxl	caption	129M	12.2	0.594	0.549	0.564	0.600	0.894	0.714	0.523	0.626	0.561
InstructBLIP [22]	ViT g-14 + Flant5xxl	list	129M	12.3	0.593	0.559	0.569	0.613	**0.897**	0.725	0.546	0.640	0.582
InstructBLIP	ViT g-14 + Flant5xxl	caption	129M	12.3	0.603	0.535	0.561	0.604	0.752	0.667	0.572	0.585	0.572
InstructBLIP	ViT g-14 + Flant5xxl	instruct	129M	12.3	0.529	**0.605**	0.556	0.569	0.881	0.686	0.559	**0.698**	0.614
Ours	ViT L-14 + $\text{Lang}_{\text{truncated}}$	-	3M	1.78	**0.673**	0.598	**0.627**	0.618	0.799	0.695	0.560	0.595	0.570
Ours	ViT L-14 + $\text{Lang}_{\text{truncated}}$	-	70M	1.78	0.659	0.577	0.609	**0.674**	0.866	**0.755**	**0.594**	0.615	**0.597**

Table A.10. **Comparison of different methods with top-5 predictions**. The same settings as in Table 1.

$p_j = \texttt{cat}$, and $\mathbb{I}(\cdot)$ is binary. This R_{ecall} is also called Exact Recall [124], also known as accuracy in image classification tasks [99]. In detail, to evaluate a classifier on ImageNet, each image has $M = 1$ GT label and $N = 1000$ class predictions, then Eq. A.6 becomes

$$\text{top-}k \text{ accuracy} = R_{ecall} = \mathbb{I}(g_1 \in \mathcal{P}_{1:k}), \tag{A.7}$$

For open-set recognition, $TP = \sum_{i=1}^{M} \mathbb{I}(g_i \in \mathcal{P})$, $g_i \in \mathcal{P}$ is a greedy matching but $\mathbb{I}(\cdot)$ is not binary because correct prediction might not be exactly the same as g_i. For instance, $g_i = \texttt{cat}$, $p_j = \texttt{kitty}$ or $\texttt{feline}$ or $\texttt{moggie}$ are all correct with high semantic similarity, and $p_j = \texttt{dog}$ or $\texttt{desk}$ are wrong with low semantic similarity. $\mathbb{I}(\cdot)$ is continuous to represent degrees of semantic similarity between g_i and p_j. One common choice for $\mathbb{I}(\cdot)$ is cosine similarity $\mathbf{S}_{ij}$ between contextual embeddings of g_i and p_j, then Eq. A.6 becomes

$$R_{ecall} = \frac{1}{M} \sum_{i=1}^{M} \max_j \mathbf{S}_{ij}, \tag{A.8}$$

which is a.k.a. BERT Recall [124]. For the open-set case, each image has $M \geq 1$ GT labels and $N \geq 1$ predictions, then top-k accuracy is

$$R_{ecall}^{top\text{-}k} = \frac{1}{M} \sum_{i=1}^{M} \mathbb{I}(g_i \in \mathcal{P}_{1:k}) = \frac{1}{M} \sum_{i=1}^{M} \max_{j \in [1,k]} \mathbf{S}_{ij}. \tag{A.9}$$

The top-k refers to the k most relevant predictions of *all possible labels in the world* to the image.

A.5. Data Preprocessing

For an image, the paired caption is preprocessed using the steps summarized in the following table.

step	details
1	Lowercase the caption.
2	Eliminate high-frequency noise words that lack meaningful content. The noise words removed in our work are [person, persons, stock, image, images, background, ounce, illustration, front, photography, day].
3	Keep only the letters, and a few special characters like spaces ( ), periods (.), commas (,), ampersands (&), and hyphens (-). Exclude all others, including numbers and words containing numbers.
4	Use NLTK [8] to tokenize the caption into words. Then tag the words with their part-of-speech (POS) tags to filter out words that are not nouns. The noun tags used in this paper are [NN, NNS].
5	Lemmatize the words to their root forms. For example, the word "dogs" is lemmatized to "dog".

With this preprocessing, we obtain a set of meaningful noun words for each image and summarize the information in the following table, including the number of image-caption pairs and distinct nouns.

statistics	CC3M train	CC3M val	COCO train	COCO val	SBU train	OpenImages val	LAION train
# images	2.69M	12478	118287	5000	828816	41686	67M
# nouns	22890	4875	15444	3834	132372	3119	2.7M

The training split contains 2,794,419 distinct nouns, while all validation splits have a total of 8,637 distinct nouns. The

number of overlapping nouns between the training and validation splits is 8,347, which is 97.8% of distinct nouns in validation splits.

A.6. Prompt Settings

For training, we adopt the prompt augmentation, which contains different prompt templates but with the same semantic meaning. In each training iteration, we randomly select one prompt from those templates for the batched images. For inference, we only use one simple prompt in all experiments. The prompt templates are listed as follows.

setting	prompt templates
training	The objects in the image are
	The items present in the picture are
	The elements depicted in the image are
	The objects shown in the photograph are
	The items visible in the image are
	The objects that appear in the picture are
	The elements featured in the image are
	The items captured in the photograph are
	The elements seen in the picture are
	The items represented in the image are
inference	The objects in the image are

For comparison, we evaluate chat-based VQA models, i.e., BLIP-2 [65], InstructBLIP [22], and LLaVA [68, 69], with two types of prompt, which are

1) text completion: `The objects in the image are`,
2) and VQA: `Describe every detail in the image.`

We refer to the text completion prompt as *prompt: list* and the VQA prompt as *prompt: caption*. After obtaining model outputs, we apply the rule from Section A.5 to extract nouns as predicted labels.

Especially, Flamingo [1, 3] has a unique prompt setting with few-shot instruction. For the *caption* type, we change the prompt setting to `What objects are in the image?`. Then we construct the prompt with 4-shot samples as in [1], which is listed as the following tables.

the *list* prompt type with few-shot samples for Flamingo
<image>The objects in the image are boy, bush, chair, clothes, grass, house, tree, sports ball.<|endofchunk|> <image>The objects in the image are bus, car, clouds, house, leaves, person, road.<|endofchunk|> <image>The objects in the image are giraffe, grass, tree.<|endofchunk|> <image>The objects in the image are cat, telecontroller, sofa.<|endofchunk|> <image>The objects in the image are

the reference images as few-shot samples for Flamingo

# tokens	CC3M R	CC3M P	CC3M F_1	COCO R	COCO P	COCO F_1	OpenImages R	OpenImages P	OpenImages F_1
prompt: list									
64	0.542	0.556	0.540	0.482	0.842	0.606	0.455	0.622	0.518
128	0.544	0.557	0.542	0.494	0.871	0.623	0.476	0.641	0.538
256	0.542	0.556	0.540	0.482	0.842	0.606	0.455	0.622	0.518
prompt: caption									
64	0.601	0.539	0.561	0.600	0.893	0.714	0.523	0.626	0.562
128	0.609	0.539	0.561	0.600	0.893	0.714	0.523	0.626	0.562
256	0.600	0.539	0.560	0.601	0.894	0.714	0.512	0.643	0.562

Table A.11. **Different number of sampling tokens** for BLIP-2 with top-10 predictions.

# tokens	CC3M R	CC3M P	CC3M F_1	COCO R	COCO P	COCO F_1	OpenImages R	OpenImages P	OpenImages F_1
prompt: list									
256	0.596	0.554	0.567	0.613	0.897	0.725	0.546	0.640	0.582
512	0.596	0.554	0.567	0.613	0.897	0.725	0.544	0.634	0.578
prompt: caption									
256	0.639	0.487	0.546	0.690	0.662	0.673	0.647	0.539	0.581
512	0.639	0.487	0.546	0.690	0.662	0.673	0.647	0.539	0.581

Table A.12. **Different number of sampling tokens** for InstructBLIP with top-10 predictions.

A.7. Number of Sampling Tokens in Comparison

We have various models to compare with ours. For a fair comparison, we need to take care of the maximum number of sampling tokens for each model to make sure that we can extract enough potential nouns words from their outputs. LLaVA [68, 69] has a maximum number of sampling tokens of 1024, which is already enough for the task. BLIP-2 [65] has a maximum 32 in default, but we change it to 64 for top-5 and 128 for top-10. To verify this setting is fair for BLIP-2, we ablate the number of sampling tokens for BLIP-2 with the caption prompt in Table A.11. For InstructBLIP [22], we use its default number of sampling tokens, which is 256 for top-5 and top-10. To verify the setting, we ablate the number of sampling tokens for InstructBLIP in Table A.12. Due to Flamingo [1, 3] has the same output format as ours, we keep the same maximum number of sampling tokens for it as ours for greedy search, i.e., 64 for top-5. We double the number to 128 for its top-10 predictions. For VQA methods, sampling more tokens for more potential predictions significantly increases time cost, esp. with beam search.

A.8. Visualizing Predictions

We visualize the top-10 predictions from our 1.78B model trained on G70M in Figure A.3-A.9 without cherry-picking. The image is paired with two columns: our predictions on the left, probability-indicating ranking bars on the right. The images sampled from COCO have gray column to show GPT-4V Preview's [86] predictions, intuitively illustraing the strengths and weaknesses of our method with the apples-to-apples comparison.

A.9. Discussion

In this section, we discuss the limitations of our method and experiments that we have tried but does not work well.

Less Is More. Our method's performance heavily relies on the quality of the training data. More noisy data will hurt the performance, for example, models trained on the noisier CC12M [12] underperform compared to those trained on CC3M [104]. Moreover, high quality requires more human efforts, which is expensive, meaning to densely annotate all possible labels for each image. We might consider using GPT-4V [86] for generating high-quality labels, though it may be costly (API expenses) and subject to the hallucination issue. Exploring methods to train models with fewer labels for broader generalization could be intriguing.

Defining Labels. How to define the label for describing an object in an image? A label could be a word, which is used in this paper, but also could be a phrase or a sentence. We have tried to define the label with the noun phrase, which includes an adjective, for example, "gray car" and "cute boy". However, these models underperformed, partly due to poor training data quality and the limitations of the parser for extracting noun phrases from captions. We also experimented with concrete nouns for training, but the results were unsatisfactory due to noisy reference labels produced by the parser, which needs a comprehensive filter to remove noise.

Evaluation. First, our evaluation has limitations due to the incomplete and imperfect nature of reference labels derived from raw captions. Second, we calculate P, R and F_1 score based on the semantic similarity between the embeddings of predicted and reference labels from a pretrained language model. However, such a model-based semantic similarity brings noise and bias to the evaluation results due to the model imperfection. This motivates us to conduct the cross-validation experiments in Section A.1, which views GPT-4V's [86] predictions as reference labels. Developing a reliable evaluation metric beyond human evaluation or model-based semantic similarity is an interesting topic.

Fine-Grained Recognition. Our method, though not designed for fine-grained recognition, could be adapted for such tasks. Currently, the method underperforms in this area due to the use of general, rather than fine-grained, training data. Improving performance may be possible by using more specific, fine-grained training data, which circles back to the initial question regarding the quality of training data.

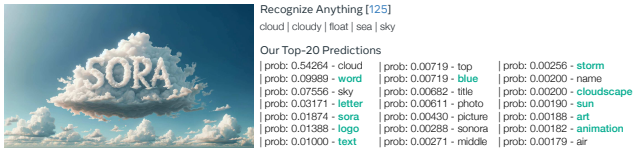
Single-Label Prediction. Our method is optimized for top-k predictions and exhibits lower performance in top-1 accuracy evaluations. Our approach encourages the model to predict multiple labels for an image, which is more realistic than predicting just one label because images generally contain multiple objects. Therefore, we do not focus on improving top-1 accuracy in this paper.

Competition Issue. We acknowledge the inherent competitive issue in our one-shot sampling, similar to the repetition issue observed in sequence-based methods like greedy and beam search. However, its results are still promising in experiments, which is likely due to redundant tokenization. Mitigating or analyzing the competition issue for the one-shot sampling could be our future research topic.

A.10. Other Related Works

Approaching object recognition as a natural language prediction, pioneered by [4, 31, 85], has been proposed before the deep learning era [63]. The motivation is primarily to assist journalists in annotating images for retrieval purposes [5, 79]. [85] slices an image into regions and predicts words using probabilistic models. [31] views recognition as a machine translation problem, aligning image regions with words using a lexicon, optimized by the EM algorithm [24].

Image Annotation and Multi-label Prediction. The evolution of image annotation or tagging closely mirrors that of multi-label prediction. Initial approaches develop on topic models [53] like latent Dirichlet allocation [5] and probabilistic latent semantic analysis [49, 84]. Mixture models [32, 52, 62] have also been explored to model the joint distributions over images and tags. Then SVM-based discriminative models [21, 47, 54] are proposed to predict tags. Later, the annotation task is treated as a retrieval problem [39, 76] based on nearest neighbors [20] or joint optimization [13]. The difficulty of collecting multi-label annotations inspires curriculum learning-based models [18, 30] and semi-supervised methods [33, 101, 107]. Now models with ranking-based losses [37] and transformer-based architecture [51, 71, 98, 125] are introduced for tagging images, but they are still closed-set recognition models trained on heavily-annotated/cleaned datasets. In contrast, our method is an open-set recognition model trained on raw data, which is at the real open-level with a large-scale prediction capability (top-100). In the figure below, our model correctly predicts the wild terms such as `sora`, `cloudscape`, `text`, `logo`, `letter`, `art`, and `animation`, assigning probabilities for ranking or filtering, while [125] does not.



A.11. Acknowledgements

We thank Alessandro Conti, the primary author of CaSED [19], for supplying the text embedding galleries for CC3M, COCO, SBU, and LAION-400M. We also thank Damian Gessler for the help on downloading training datasets and solving cluster issues, and our group colleagues at Meta for the helpful discussions.

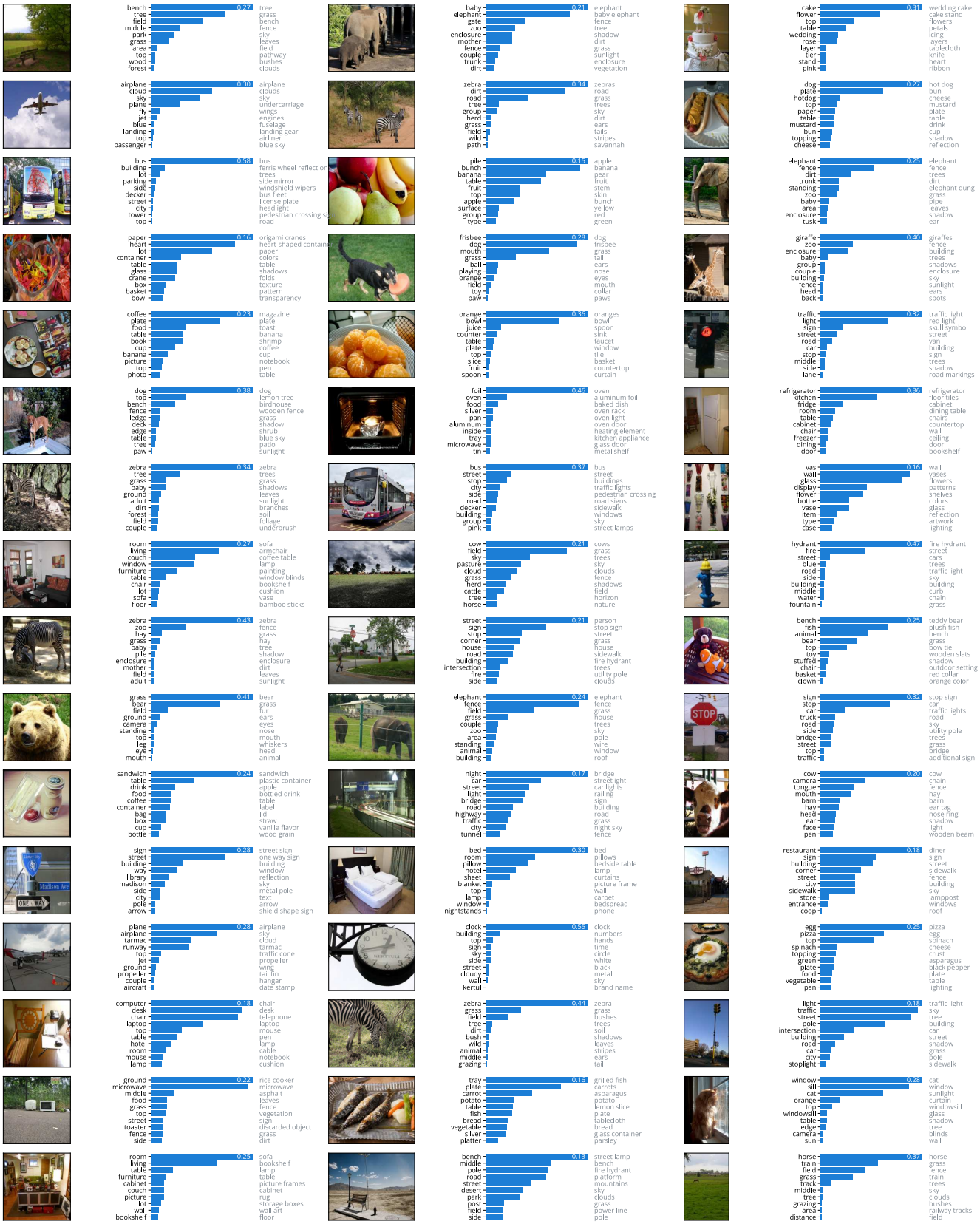


Figure A.3. Top-10 predictions on COCO validation split without cherry-picking. The top bar is with the first prediction's probability. The right column shows predictions in gray from the GPT-4V Preview. Images are licensed under a Creative Commons Attribution 2.0 License.

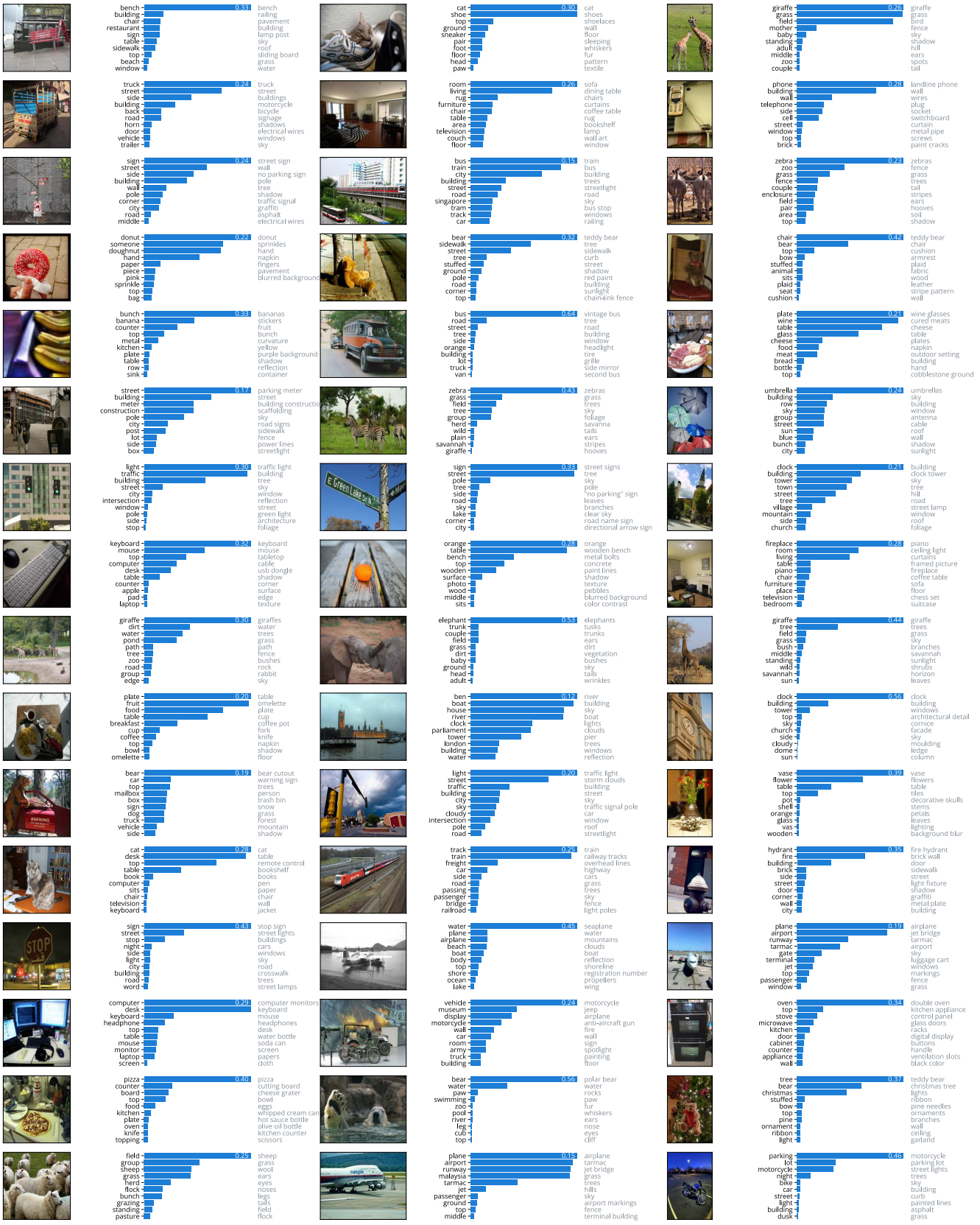


Figure A.4. Top-10 predictions on COCO validation split without cherry-picking. The top bar is with the first prediction's probability. The right column shows predictions in gray from the GPT-4V Preview. Images are licensed under a Creative Commons Attribution 2.0 License.

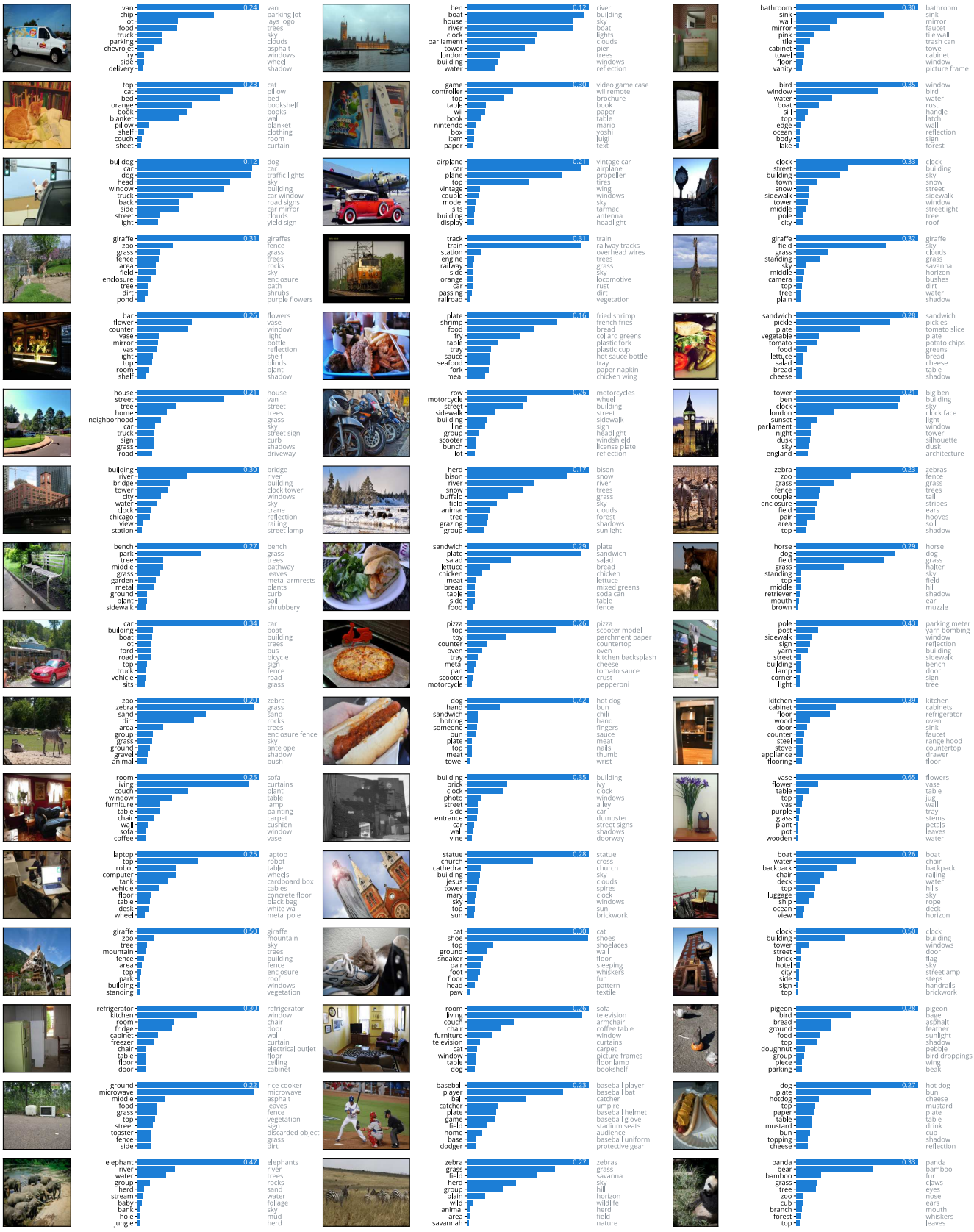


Figure A.5. Top-10 predictions on COCO validation split without cherry-picking. The top bar is with the first prediction's probability. The right column shows predictions in gray from the GPT-4V Preview. Images are licensed under a Creative Commons Attribution 2.0 License.

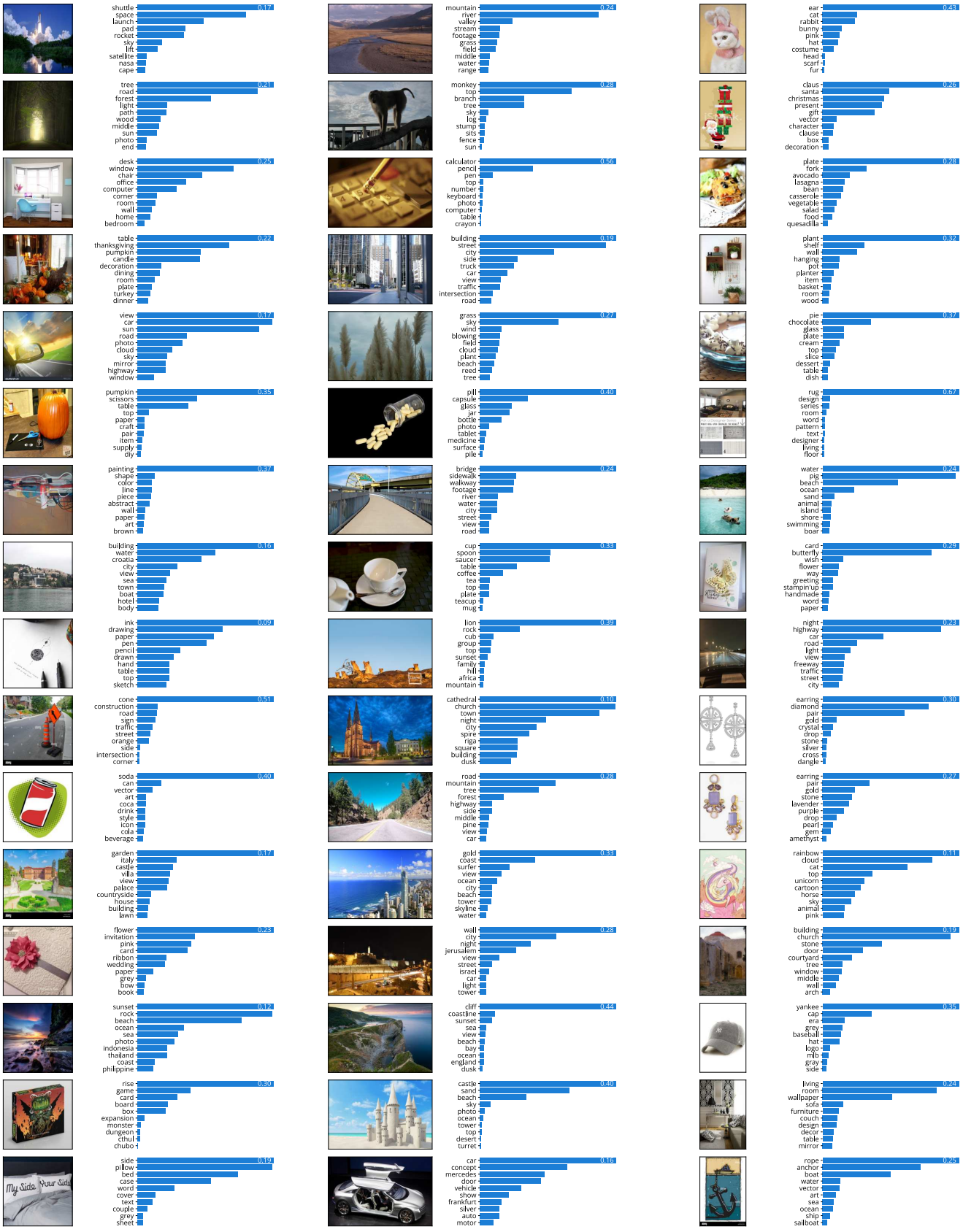


Figure A.6. Top-10 predictions on CC3M validation split without cherry-picking. The top bar is with the first prediction's probability. Images in the dataset of CC3M are provided by Google LLC.

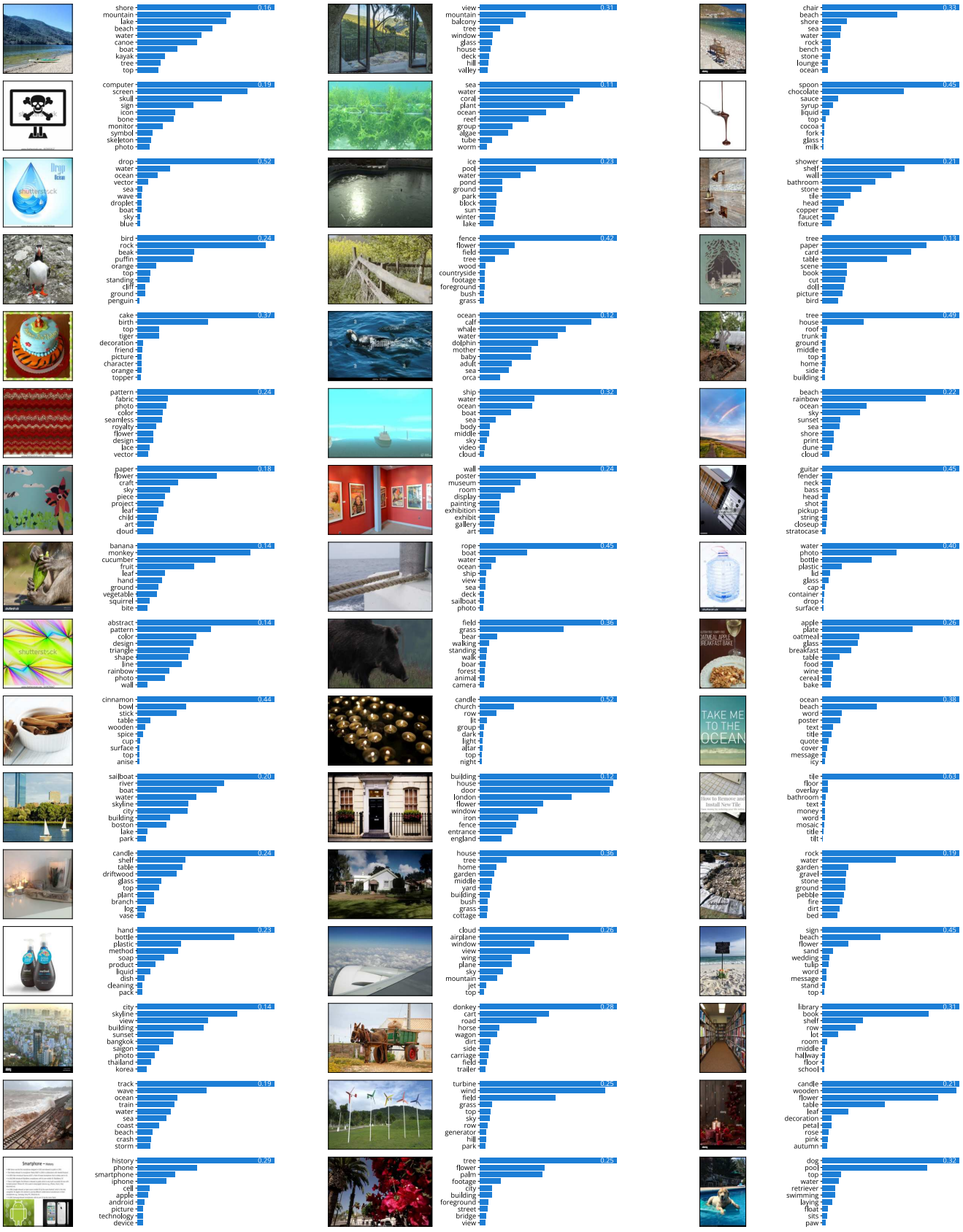


Figure A.7. Top-10 predictions on CC3M validation split without cherry-picking. The top bar is with the first prediction's probability. Images in the dataset of CC3M are provided by Google LLC.

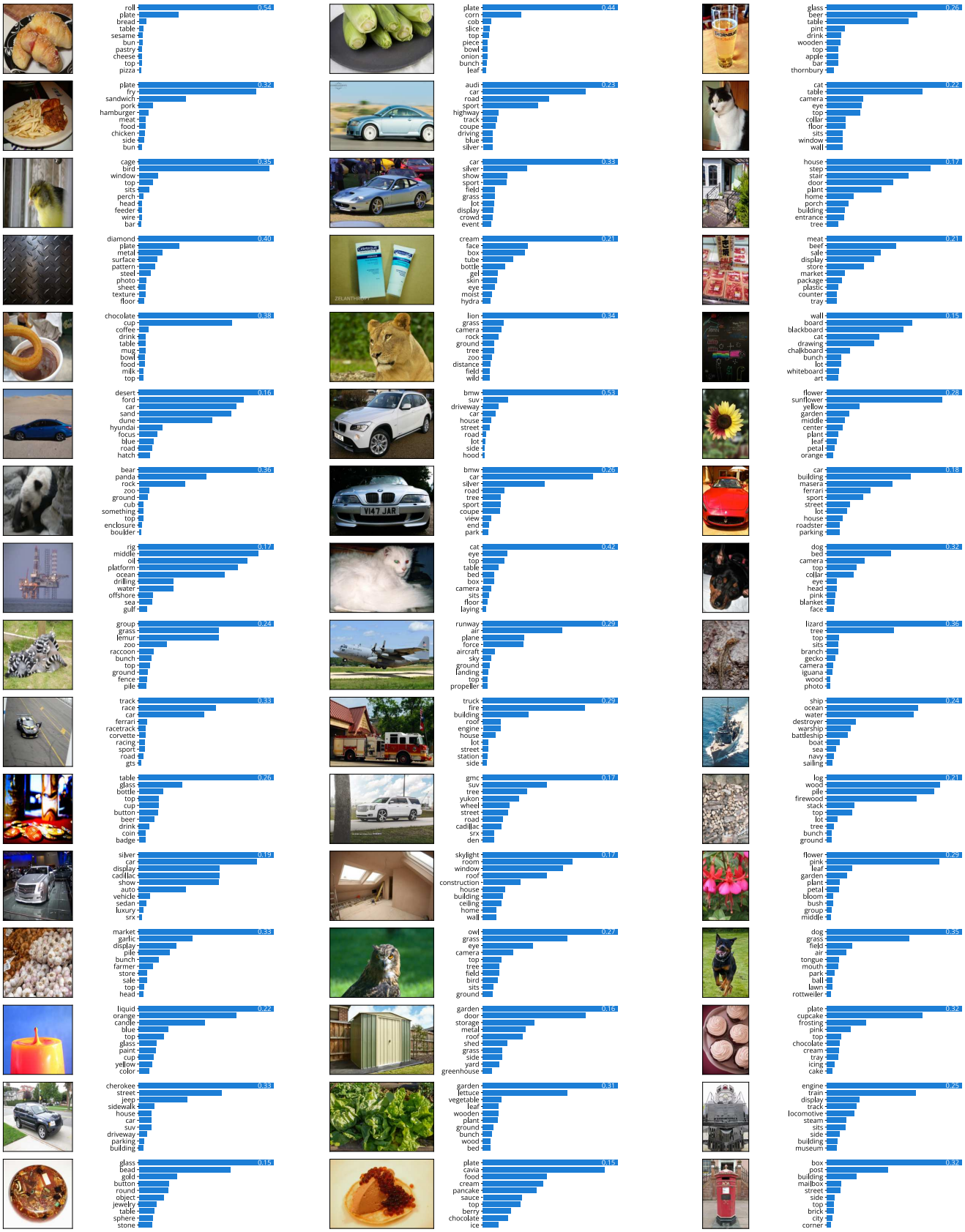


Figure A.8. Top-10 predictions on OpenImages validation split without cherry-picking. The top bar is with the first prediction's probability. Images in the dataset of OpenImages are under a Creative Commons Attribution 2.0 License.

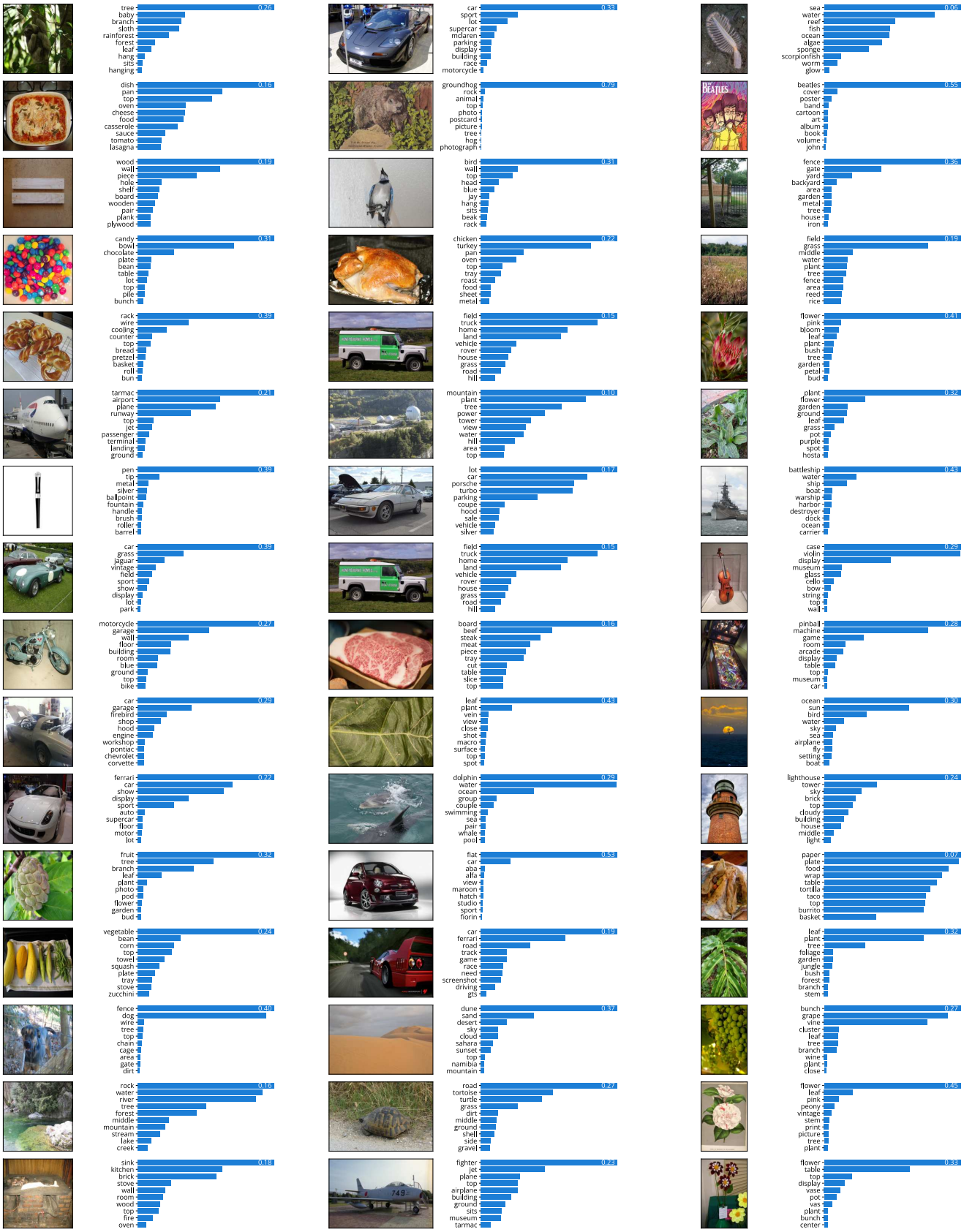


Figure A.9. Top-10 predictions on OpenImages validation split without cherry-picking. The top bar is with the first prediction's probability. Images in the dataset of OpenImages are under a Creative Commons Attribution 2.0 License.