A Appendix

The Appendix provides a Model Card [27] for OWLv2 as well as additional methodological details, hyperparameters, and results. At the end of the Appendix, we provide qualitative examples of the self-training data and model predictions.

The Appendix is structured as follows:

A.1 Model Card

Model Summary	
Model Architecture	OWL v2 is an open-vocabulary object detector based on OWL-ViT [26]. It consists of an image encoder with a Vision Transformer [17] architecture, a text encoder with a similar Transformer architecture, and heads that predict bounding boxes and label scores from provided images and text queries.
Input(s)	An image and a list of free-text object descriptions (queries).
Output(s)	A list of bounding boxes and a score for each box/query pair.
Usage	
Application	The model is intended for open-vocabulary object detection.
Known Caveats	(1) Confidence scores of predictions are not intended to be compared across text queries. While the training loss encourages cross-query calibration for *seen* queries, scores for *unseen* queries are not calibrated. Further, the mean Average Precision (mAP) metric does not measure cross-query calibration, so higher mAP does not imply better cross-query calibration. Also see Section 5. (2) Fine-tuning the model creates a trade-off between the performance on fine-tuned texts and unseen texts. See Section 4.6 for details.
System Type	
System Description	This is a standalone model.
Upstr. Dependencies	None.
Downstr. Dependencies	None.
Implementation Frameworks	
Hardware & Software	Hardware: TPU [13] v2 or v3 (for B- and L-sized models) or v4 (for G-sized models). Software: JAX [3], Flax [11], Scenic [7].
Compute Requirements	Reported in Section 4.5.
Model Characteristics	
Model Initialization	The model is initialized from pre-trained language CLIP [30] or SigLIP [43] checkpoints.
Model Status	This is a static model trained on an offline dataset.
Model Stats	The largest OWLv2 model has 2.3B parameters, of which 2B are used for the image encoder and 300M for the text encoder (the heads have a negligible number of parameters). We also trained models with 430M and 150M parameters.
Data Overview	
Training dataset	The model is self-trained on bounding boxes predicted by the original OWL-ViT L/14 model [26] on the WebLI dataset [4]. Details on the annotation procedure are provided in Section 3.1.
Evaluation & Fine-tuning Dataset	Open-vocabulary object detection performance is evaluated using the LVIS [10] and ODinW13 [21] datasets. As indicated in Table 1, some models are fine-tuned on the "base" annotations of LVIS, i.e. only annotations for "frequent" and "common" object categories as defined in the official annotations [10]. None of our models have seen any human annotations for LVIS "rare" categories, such that LVIS mAP_{rare} measures zero-shot performance.
Evaluation Results	
Evaluation Results	Reported in Table 1.
Model Usage & Limitations	
Sensitive Use	The model detects objects matching free-text descriptions. This capability should not be used for unethical use cases such as surveillance.
Known Limitations	Reported in Section 5.
Ethical Considerations	Reported in Section 5.

A.2 Human-Curated Label Space

The human-curated label space was obtained by merging common dataset class lists with the Python code below.

```
# Dataset class names, as available e.g. from TensorFlow Datasets.
# For Visual Genome, we used the 1600 most common label strings.
LVIS_CLASS_NAMES = [...]
OBJECTS365_CLASS_NAMES = [...]
OPEN_IMAGES_V4_BOXABLE_CLASS_NAMES = [...]
VISUAL_GENOME_CLASS_NAMES = [...]

queries = (
    LVIS_CLASS_NAMES
    + OBJECTS365_CLASS_NAMES
    + OPEN_IMAGES_V4_BOXABLE_CLASS_NAMES
)

# Remove duplicates:
queries = set([q.lower() for q in queries])

# Remove plural forms:
remove = set()
for singular in queries:
  plurals = [singular + 's', singular + 'es']
  for plural in plurals:
    if plural in queries:
      remove.add(plural)

# Same queries for all images:
queries = list(queries.difference(remove))
```

A.3 Machine-Generated Label Space

The machine-generated label space was obtained from the image-associated text, for each image separately, using the Python code below. Figure A3 shows example pseudo-annotations using the N-gram label space.

```
from typing import Iterable, List
import nltk

# Stopwords from nltk.corpus.stopwords.words('english'):
STOPWORDS_EN = frozenset({
    'a', 'about', 'above', 'after', 'again', 'against', 'all', 'am', 'an',
    'and', 'any', 'are', 'as', 'at', 'be', 'because', 'been', 'before', 'being',
    'below', 'between', 'both', 'but', 'by', 'can', 'did', 'do', 'does',
    'doing', 'don', 'down', 'during', 'each', 'few', 'for', 'from', 'further',
    'had', 'has', 'have', 'having', 'he', 'her', 'here', 'hers', 'herself',
    'him', 'himself', 'his', 'how', 'i', 'if', 'in', 'into', 'is', 'it', 'its',
    'itself', 'just', 'me', 'more', 'most', 'my', 'myself', 'no', 'nor', 'not',
    'now', 'of', 'off', 'on', 'once', 'only', 'or', 'other', 'our', 'ours',
    'ourselves', 'out', 'over', 'own', 's', 'same', 'she', 'should', 'so',
    'some', 'such', 't', 'than', 'that', 'the', 'their', 'theirs', 'them',
    'themselves', 'then', 'there', 'these', 'they', 'this', 'those', 'through',
    'to', 'too', 'under', 'until', 'up', 'very', 'was', 'we', 'were', 'what',
    'when', 'where', 'which', 'while', 'who', 'whom', 'why', 'will', 'with',
    'you', 'your', 'yours', 'yourself', 'yourselves'
})

# These words were found by manually going through the most common 1000 words
# in a sample of alt-texts and selecting generic words without specific meaning:
COMMON_GENERIC_WORDS = frozenset({
    'alibaba', 'aliexpress', 'amazon', 'available', 'background', 'blog', 'buy',
```

```
    'co', 'com', 'description', 'diy', 'download', 'facebook', 'free', 'gif',
    'hd', 'ideas', 'illustration', 'illustrations', 'image', 'images', 'img',
    'instagram', 'jpg', 'online', 'org', 'original', 'page', 'pdf', 'photo',
    'photography', 'photos', 'picclick', 'picture', 'pictures', 'png', 'porn',
    'premium', 'resolution', 'royalty', 'sale', 'sex', 'shutterstock', 'stock',
    'svg', 'thumbnail', 'tumblr', 'tumgir', 'twitter', 'uk', 'uploaded', 'vector',
    'vectors', 'video', 'videos', 'wallpaper', 'wallpapers', 'wholesale', 'www',
    'xxx', 'youtube'
})

def _is_all_stopwords(ngram: Iterable[str]) -> bool:
  return set(ngram).issubset(STOPWORDS_EN)


def _get_ngrams(
    caption: str, max_num_queries: int, max_ngram_len: int
) -> List[str]:
  """Returns image caption ngrams as queries."""

  # Make lower-case:
  caption = caption.lower()

  # Remove common generic words:
  words = [w for w in caption.split() if w not in COMMON_GENERIC_WORDS]

  queries = []
  for ngram in nltk.everygrams(words, max_len=max_ngram_len):
    # Don't use ngram if it only consists of stop words:
    if _is_all_stopwords(ngram):
      continue
    queries.append(' '.join(ngram))
    if len(queries) == max_num_queries:
      break
  return queries

# Example command to get queries for one image:
queries = _get_ngrams(caption, max_num_queries=300, max_ngram_len=10)
```

A.4 Combined Label Space

When merging pseudo-annotations obtained with human-curated and machine-generated queries, it is important to consider that human-curated queries tend to be closer to the training distribution of the annotator and therefore tend to have higher scores than pseudo-annotations based on machine-generated queries. Simply merging annotations from the two label spaces and filtering them with the same confidence threshold would therefore retain primarily annotations based on human-curated queries. To achieve a more even balance when using the combined label space ("N-grm+curated" in Table 1), we therefore re-scaled scores of pseudo-annotations obtained with the human-curated queries by a factor of 0.3 before applying the same confidence threshold to all (human-curated and machine-generated) annotations.

A.5 Augmentations for Self-Training

Since Web-scale image-text data differs in important aspects from human-curated detection datasets, we depart from the augmentation strategy of [26] in several ways. As described in Section 3.2, since Web images tend to be smaller and show fewer objects than e.g. LVIS images, we use stronger image mosaics with up do 6×6 tiles (Figure A1). For the same reason, we additionally randomly resize each raw image such that its width is between $0.5\times$ and $1.0\times$ the width of the full mosaic tile, padding on the bottom and right to preserve the aspect ratio (Figure A4).

On the other hand, given the large size of our dataset, some other augmentations can be avoided: We do not use left/right flipping or random cropping during self-training. We also do not add random

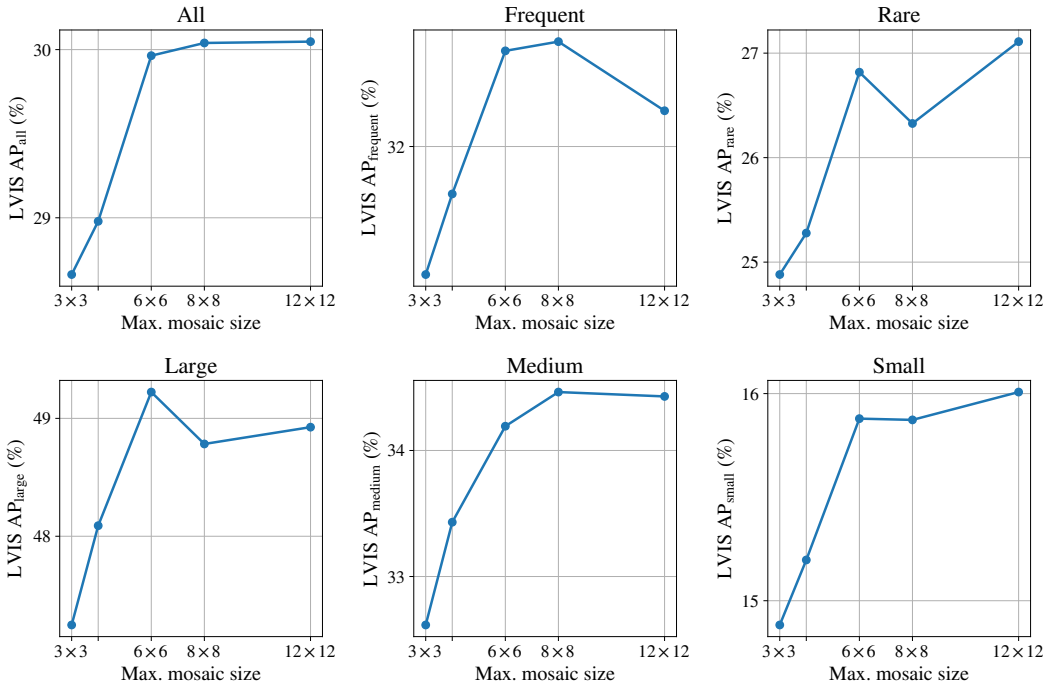


Figure A1: Sweep over mosaic sizes. OWL-ViT B/16 models were trained on pseudo-box annotations ("ngrams" label space) for 100'000 steps with different mosaic sizes. At a given "max. mosaic size", the model is trained on equal proportions of mosaics up to that size. For example, for max. size $= 12 \times 12$, the model receives images with 1, 2^2, 3^2, 4^2, 6^2, 8^2, or 12^2 tiles, respectively (only sizes with prime factors 1, 2, and 3 are supported). For this figure, the model input resolution was 768×768. Mosaic sizes up to 12×12 improve overall performance (mAP$_{all}$) and especially "rare" and "small" object performance. The benefit may be due to seeing smaller objects on average, or due to seeing more WebLI images per training step (a 12×12 mosaic contains 144 WebLI images).

prompt templates to the pseudo-labels during self-training. During fine-tuning, we use the same augmentations as [26].

A.6 Token Dropping

To improve training efficiency, we drop image patches based on their pixel variance (Section 3.2). Table A2 shows how the performance of a standard OWL-ViT model varies for different amounts of token dropping. Dropping up to 50% of tokens is within one standard deviation of the full performance. We therefore drop 50% of tokens during all of our experiments.

Table A2: Performance of standard OWL-ViT (L/14), trained on Objects365 and Visual Genome as in [26], for different token drop rates. For drop rate 0.0, the standard deviation over three runs is given.

	Token drop rate				
Metric	0.00	0.25	0.33	0.50	0.70
LVIS AP$^{val}_{all}$	33.3 $\pm$0.33	33.1	33.6	32.9	30.4
LVIS AP$^{val}_{rare}$	31.8 $\pm$1.16	31.0	32.6	30.8	28.2

To inject some stochasticity to the patch selection, we add a small amount of noise to the image before computing patch variance (uniformly distributed between 0.0 and 0.01 for images in the range $[0.0, 1.0]$). Figure A4 shows an example training image before and after token dropping.

Table A3: Hyperparameters of the models shown in Table 1. Only parameters that vary between models are shown; constant parameters are described in the text (Appendix A.8). For *Dropout rate* and *Droplayer rate*, the first number indicates the value used for the image encoder and the second for the text encoder. *Examples seen* includes both self-training and fine-tuning.

	Method	Backbone	Image size	Learning rate	Dropout rate	Droplayer rate	Instance top k	Batch size (ST)	Batch size (FT)	Examples seen
Open vocabulary:										
11	OWL-ST	CLIP B/16	960	5×10^{-5}	.0/.0	.2/.1	256	256	–	3.7×10^8
12	OWL-ST	CLIP L/14	1008	2×10^{-5}	.0/.0	.2/.1	512	256	–	2.3×10^8
13	OWL-ST	SigLIP G/14	1008	2×10^{-5}	.0/.1	.2/.4	512	128	–	1.6×10^8
14	OWL-ST+FT	CLIP B/16	960	5×10^{-5}	.0/.0	.2/.1	256	256	256	3.6×10^8
15	OWL-ST+FT	CLIP L/14	1008	2×10^{-5}	.0/.0	.2/.1	512	256	128	2.3×10^8
16	OWL-ST+FT	SigLIP G/14	1008	2×10^{-5}	.0/.1	.2/.4	512	128	128	1.6×10^8
Human-curated vocabulary:										
20	OWL-ST+FT	CLIP B/16	960	5×10^{-5}	.0/.0	.2/.1	256	256	256	8.2×10^8
21	OWL-ST+FT	CLIP L/14	1008	2×10^{-5}	.0/.0	.2/.1	512	256	128	3.6×10^8

A.7 Further Efficiency Improvements

To further improve training efficiency beyond the methods described in Section 3.2, we also adopt previously proposed methods for large-scale Transformer training: To save memory, we use a variant [42] of the Adafactor optimizer [34] instead of Adam [15]. To avoid having to choose and optimize the total training duration ahead of time, we use the open-ended inverse square-root schedule [36, 42] with a fixed time-scale of 10'000 steps for all experiments and linearly "cool down" checkpoints along the way for evaluation (see Section 3.3).

A.8 Model Hyperparameters

We use the following hyperparameters for all of our models. Hyperparameters that vary between models are listed in Table A3.

- Optimizer: Adafactor variant as in [42]
- Learning rate schedule: Inverse square-root [36] with timescale 10'000 steps
- Learning rate for the text encoder: 2×10^{-6}
- Token dropping rate during training: 0.5
- Pseudo-annotation confidence score threshold: 0.3 (except for Figure 3)
- Augmentations: See Appendix A.5
- All remaining hyperparameters are as in [26].

Hyperparameter selection. Most hyperparameters were either taken directly from [26] or technically constrained, e.g. we chose the largest batch size that fit into the memory of the available accelerators. Where hyperparameters were tuned, we ran short B/16-scale trial experiments and selected the parameters with the highest LVIS mAP_{rare} for our main runs.

SigLIP G/14. For the G/14 model, we started self-training with a learning rate of 5×10^{-5}, a droplayer rate of .1/.0, and no dropout. We found that the model overfit during fine-tuning with these settings, and switched to a learning rate of 2×10^{-5}, a droplayer rate of .2/.4, and a dropout rate of .0/.1 after 740'000 self-training steps. To save resources, we did not start training from the beginning. With the new settings, we observed no overfitting during fine-tuning, but it is possible that these settings are still not optimal.

Table A4: Open-vocabulary detection results on LVIS using the "fixed" AP metric [5]. Fixed AP is implemented as proposed in [5] by evaluating AP on the top 10'000 predictions per class over the entire validation set.

	Method	Backbone	AP^{mini}_{all}		AP^{mini}_{rare}		AP^{val}_{all}		$\mathbf{AP^{val}_{rare}}$	
			old	fixed	old	fixed	old	fixed	old	fixed
Open vocabulary:										
1	RegionCLIP [46]	R50x4	–	–	–	–	32.3	–	22.0	–
2	OWL [26]	CLIP B/16	–	–	–	–	27.2	–	20.6	–
3	OWL [26]	CLIP L/14	–	–	–	–	34.6	–	31.2	–
4	GLIPv2 [45]	Swin-T	29.0	–	–	–	–	–	–	–
5	GLIPv2 [45]	Swin-B	48.5	–	–	–	–	–	–	–
6	GLIPv2 [45]	Swin-H	50.1	–	–	–	–	–	–	–
7	F-VLM [19]	R50x4	–	–	–	–	28.5	–	26.3	–
8	F-VLM [19]	R50x64	–	–	–	–	34.9	–	32.8	–
9	3Ways [1]	NFNet-F0	–	–	–	–	35.7	–	25.6	–
10	3Ways [1]	NFNet-F6	–	–	–	–	44.6	–	30.1	–
11	OWL-ST	CLIP B/16	31.8	34.4	35.4	38.3	27.0	28.6	29.6	30.3
12	OWL-ST	CLIP L/14	38.1	40.9	39.0	41.5	33.5	35.2	34.9	36.2
13	OWL-ST	SigLIP G/14	37.8	–	40.9	–	33.7	–	37.5	–
14	OWL-ST+FT	CLIP B/16	47.2	48.7	37.8	42.1	41.8	43.2	36.2	39.0
15	OWL-ST+FT	CLIP L/14	54.1	56.2	46.1	52.3	49.4	51.1	44.6	47.4
16	OWL-ST+FT	SigLIP G/14	51.3	–	50.9	–	47.0	–	47.2	–
Human-curated vocabulary:										
17	Detic [47]	R50	–	–	–	–	32.4	–	24.6	–
18	DetCLIPv2 [38]	Swin-T	–	40.4	–	36.0	–	32.8	–	31.0
19	DetCLIPv2 [38]	Swin-L	–	44.7	–	43.1	–	36.6	–	33.3
20	OWL-ST+FT	CLIP B/16	51.1	52.3	41.9	46.5	45.6	46.7	40.5	42.5
21	OWL-ST+FT	CLIP L/14	55.8	57.2	50.0	54.5	50.4	52.0	45.9	48.5

A.9 Additional Results

A.9.1 Fixed Average Precision

In the standard Average Precision metric (AP^{old}), performance on one class depends on the performance on other classes. This dependence makes the metric "gameable" by re-scaling the scores of certain classes [5]. To avoid this issue, some prior work reports a "fixed" version of AP proposed in [5]. In Table 1, we report AP^{old} for our models. For models from the literature, we report whichever AP version is available. Since AP^{fixed} tends to produce higher values than AP^{old}, Table 1 tends to underestimate the advantage of our method over prior work using AP^{fixed}. We provide AP^{fixed} for all of our models in Table A4. As proposed in [5], we implement AP^{fixed} by evaluating AP on the top 10'000 predictions per class over the entire validation set. This ensures that classes do not compete with each other for inclusion in the evaluated predictions.

A.9.2 Per-Dataset ODinW Results

Table A5 shows un-aggregated results on all 35 ODinW datasets for our main models. In addition, in the last row, we provide results for a weight-space ensemble of a self-trained and fine-tuned OWLv2 L/14 model (the same model is shown in Figure A2).

A.9.3 Fine-Tuning Robustness Trade-Off for OWLv2 L/14

In Figure A2, we provide the same analysis of the robustness trade-off after fine-tuning for an L/14 model that we provided for a B/16 model in Figure 5.

A.10 Qualitative Examples

In Figures A5 to A7, we provide qualitative examples of detection predictions from OWLv2 L/14 models. In each figure, the top image shows predictions obtained directly after self-training, and

the bottom image shows predictions after fine-tuning on $\text{LVIS}_{\text{base}}$. Example images are from the LVIS validation set and the model was queried with all LVIS classes. All predictions meeting the confidence threshold specified in the caption are shown.

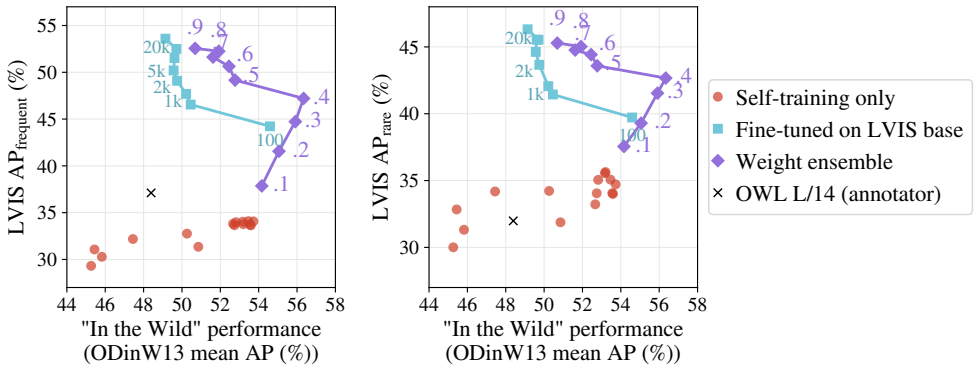


Figure A2: Trade-off between fine-tuned and open-world performance. Similar to Figure 5, but for OWLv2 L/14.

Table A5: Zero-shot AP of the models in Table 1 on all 35 ODinW datasets [21]. The subset of 13 datasets defined in [22] and used in the main paper is shown in **bold**. The last row (*OWL-ST/FT ens*) shows the weight-space ensemble [37] of the checkpoints after self-training and after fine-tuning of the model in row 21 (weight of the fine-tuned checkpoint in the ensemble is 0.4; also see Figure A2). This is our best model by ODinW13 performance.

	Method	Backbone	Mean (13 datasets)	Mean (35 datasets)	Median (35 datasets)	**Aerial Maritime Drone Large**	Aerial Maritime Drone Tiled	American Sign Language	**Aquarium**	BCCD	Boggle Boards	Brackish Underwater	Chess Pieces	**Cottontail Rabbits**	Dice Medium Color	Drone Control	**Ego Hands Generic**	Ego Hands Specific	Hard Hat Workers	Mask Wearing	Mountain Dew Commercial	**North America Mushrooms**	Open Poetry Vision	Oxford Pets By Breed	Oxford Pets By Species	**Packages**	**Pascal VOC**	**Pistols**	Pk Lot	Plantdoc	**Pothole**	**Raccoon**	Selfdriving Car	**Shellfish OpenImages**	Thermal Cheetah	**Thermal Dogs And People**	Uno Cards Raw	**Vehicles OpenImages**	Website Screenshots	Wildfire Smoke
Open vocabulary:																																								
11	OWL-ST	CLIP B/16	48.8	22.1	11.6	11.6	19.4	1.1	33.2	11.6	0.3	4.8	4.1	85.5	0.1	2.7	46.9	5.5	2.0	0.4	22.0	33.9	0.4	2.7	3.4	75.9	52.7	60.1	0.1	4.8	19.2	66.6	5.5	40.1	19.1	51.1	1.0	57.2	1.8	25.4
12	OWL-ST	CLIP L/14	53.0	24.4	16.2	19.9	21.2	1.1	32.3	16.2	0.2	5.9	7.8	84.9	0.1	4.7	47.1	3.5	1.9	0.5	27.3	76.6	0.6	3.1	2.7	70.9	53.9	62.6	0.0	4.4	27.5	63.8	4.9	35.0	25.5	55.6	1.1	58.5	1.8	31.1
13	OWL-ST	SigLIP G/14	49.9	22.9	17.5	22.0	17.5	2.0	36.7	21.4	0.2	3.3	5.6	88.1	0.1	4.9	37.8	4.3	1.4	0.2	22.6	42.4	0.5	3.0	3.2	62.8	53.4	58.4	0.1	6.5	25.7	63.9	5.8	42.5	25.0	56.6	1.2	58.1	2.0	23.4
14	OWL-ST+FT	CLIP B/16	48.6	20.8	6.0	13.7	16.6	0.2	35.8	3.9	0.1	4.2	3.1	85.5	0.1	0.9	50.7	1.3	2.7	0.5	16.0	37.4	0.2	1.9	2.1	71.3	57.4	59.4	0.2	2.7	7.6	61.7	6.0	42.5	15.3	45.6	1.3	62.8	1.2	15.8
15	OWL-ST+FT	CLIP L/14	50.1	22.3	6.3	20.6	16.3	0.2	37.4	4.0	0.1	5.1	5.6	83.4	0.1	4.8	58.5	2.2	2.1	0.6	28.5	42.2	0.3	2.5	1.9	65.5	58.9	63.7	0.2	1.5	9.1	57.2	6.3	43.0	24.7	47.7	1.3	64.3	1.8	20.3
16	OWL-ST+FT	SigLIP G/14	50.1	22.5	9.5	21.3	16.5	0.3	39.8	9.5	0.3	5.6	5.8	82.5	0.0	3.6	50.9	0.5	1.7	0.2	25.5	44.9	0.2	2.8	2.3	68.1	56.4	58.5	0.7	5.3	17.4	58.3	6.1	42.7	23.6	47.9	1.9	61.9	1.9	23.9
Human-curated vocabulary:																																								
20	OWL-ST+FT	CLIP B/16	48.9	21.7	6.8	16.7	17.2	0.3	35.3	4.5	0.1	4.6	4.4	85.1	0.1	2.4	51.8	0.9	2.9	0.4	27.3	36.9	0.3	2.1	2.5	71.3	59.0	61.3	0.4	2.7	9.6	58.7	6.8	42.0	20.0	45.7	1.2	62.6	1.5	20.6
21	OWL-ST+FT	CLIP L/14	48.7	21.9	7.0	18.8	17.5	0.2	36.4	5.3	0.1	5.4	5.7	85.1	0.1	4.9	53.9	2.5	2.2	0.3	28.8	41.2	0.3	2.4	2.1	61.1	59.2	65.7	0.1	1.8	9.5	57.9	7.0	44.0	23.8	36.8	0.9	63.2	1.6	20.7
	OWL-ST/FT ens	CLIP L/14	56.3	25.6	10.6	21.7	20.0	1.0	39.1	10.6	0.2	7.6	7.0	87.0	0.0	6.1	53.1	3.2	2.1	0.3	31.3	80.6	0.4	3.1	2.9	66.3	61.8	66.2	0.1	4.0	26.0	65.4	6.2	45.1	24.1	56.7	1.1	63.3	1.9	30.9

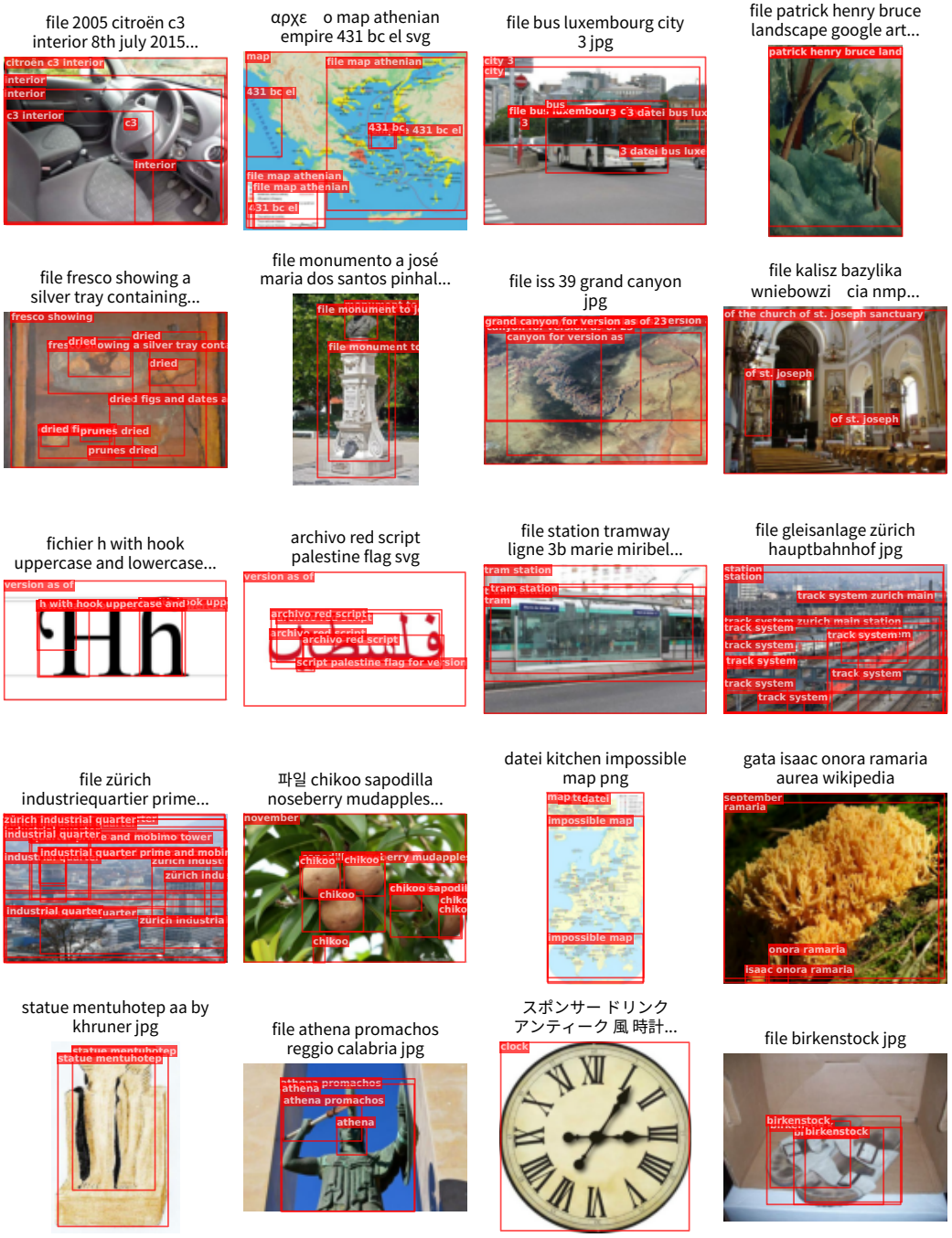


Figure A3: Example pseudo-annotations on WebLI [4]. Image-associated text (from the HTML `alt_text` tag) is shown above the images. If the text is not in English, an automatically generated translation is used. N-grams are extracted from these texts to generate queries for the annotator model. Pseudo-annotations were filtered as for our main experiments: To be included, boxes must have a score of at least 0.1, and images must have at least one box with a score above 0.3. All images from Wikimedia Commons.

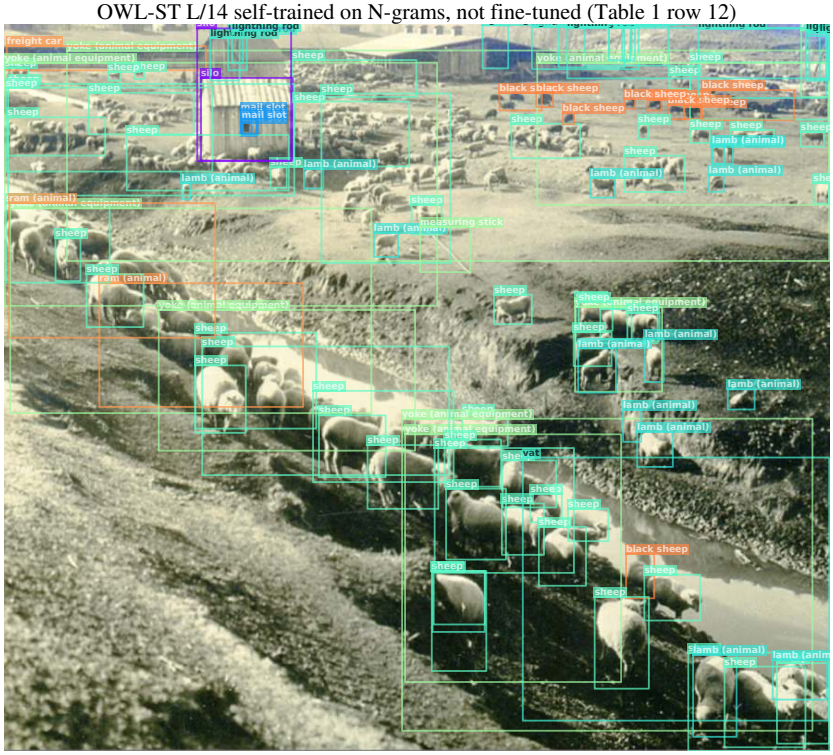
Figure A4: Training inputs after pre-processing. **Top:** A 4×4 mosaic of randomly resized and padded images as used for self-training. **Bottom:** The same mosaic after dropping the 50% of patches with lowest pixel variance (image size: 1008×1008; patch size: 14×14). Most dropped patches belong to padding areas or uniform image backgrounds. All images from Wikimedia Commons.

OWL-ST L/14 self-trained on N-grams, not fine-tuned (Table 1 row 12)

OWL-ST+FT L/14 self-trained on N-grams and fine-tuned on $\text{LVIS}_{\text{base}}$ (Table 1 row 15)

Figure A5: Qualitative example for OWLv2 L/14 from the LVIS val set. For the visualization, all LVIS classes were used as prompts. $\text{LVIS}_{\text{rare}}$ classes are labeled in black. **Top:** OWL-ST self-trained on N-grams, not fine-tuned (Table 1 row 12). **Bottom:** OWL-ST+FT self-trained on N-grams and fine-tuned on $\text{LVIS}_{\text{base}}$ (Table 1 row 15). Boxes above score 0.08 (top) or 0.3 (bottom) are shown.

OWL-ST L/14 self-trained on N-grams, not fine-tuned (Table 1 row 12)



OWL-ST+FT L/14 self-trained on N-grams and fine-tuned on $\text{LVIS}_{\text{base}}$ (Table 1 row 15)

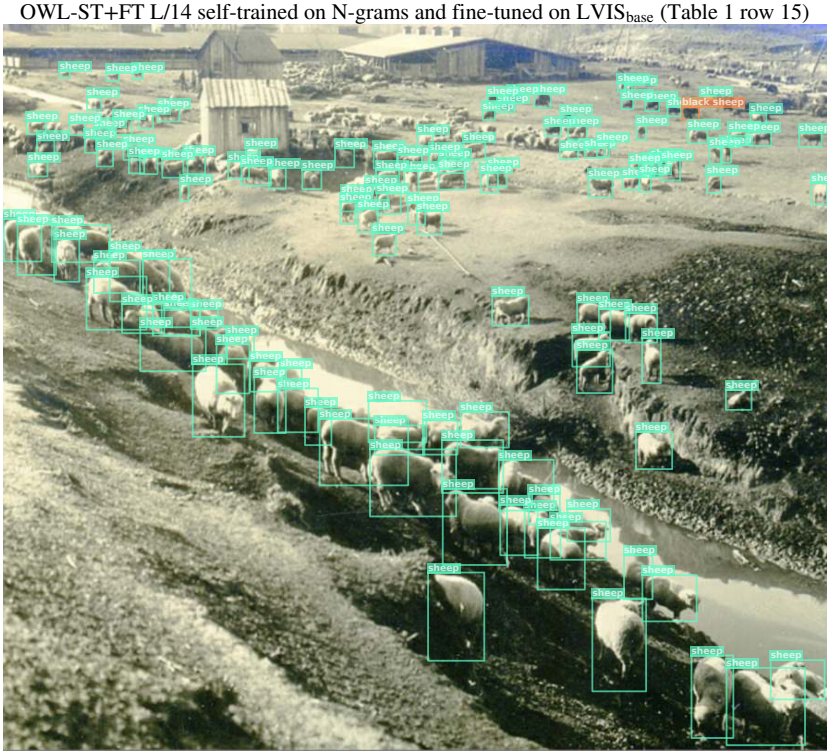
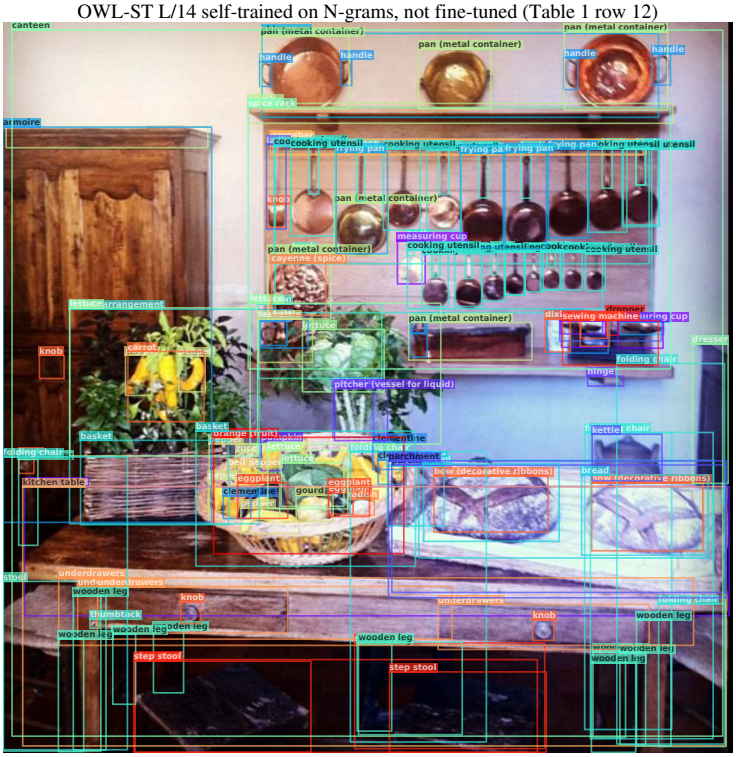


Figure A6: Qualitative example for OWLv2 L/14 from the LVIS val set. For the visualization, all LVIS classes were used as prompts. $\text{LVIS}_{\text{rare}}$ classes are labeled in black. **Top:** OWL-ST self-trained on N-grams, not fine-tuned (Table 1 row 12). **Bottom:** OWL-ST+FT self-trained on N-grams and fine-tuned on $\text{LVIS}_{\text{base}}$ (Table 1 row 15). Boxes above score 0.08 (top) or 0.3 (bottom) are shown.

OWL-ST L/14 self-trained on N-grams, not fine-tuned (Table 1 row 12)



OWL-ST+FT L/14 self-trained on N-grams and fine-tuned on $\text{LVIS}_{\text{base}}$ (Table 1 row 15)

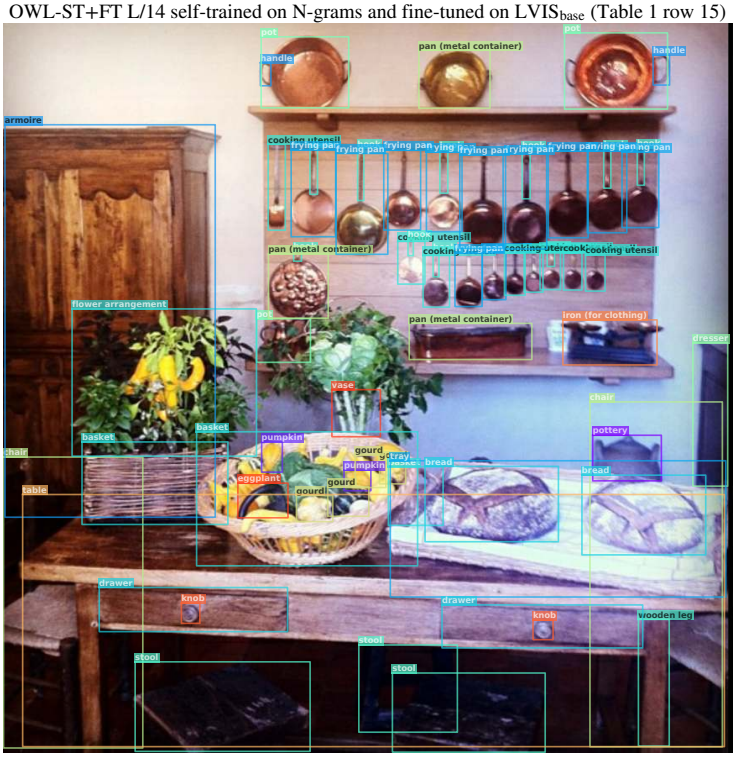


Figure A7: Qualitative example for OWLv2 L/14 from the LVIS val set. For the visualization, all LVIS classes were used as prompts. $\text{LVIS}_{\text{rare}}$ classes are labeled in black. **Top:** OWL-ST self-trained on N-grams, not fine-tuned (Table 1 row 12). **Bottom:** OWL-ST+FT self-trained on N-grams and fine-tuned on $\text{LVIS}_{\text{base}}$ (Table 1 row 15). Boxes above score 0.08 (top) or 0.3 (bottom) are shown.