Seeing more with less: human-like representations in vision models

Supplementary Material

1. Sampling scheme

In this section, we provide more detailed information on the particular sampling schemes we utilized for our S_{uni}, S_{var}, and S_{baseline} functions in the main paper.

Baseline. We refer to the given original images, without any pre-processing, as "baseline" or "full resolution," utilizing 100% of the available pixel budget. The FOV spanned by the original image pixels is referred to as the "full FOV." The function S_{baseline} is then defined as sampling all pixels (the final image remains unchanged):

$$S_{\text{baseline}}(x, y) = 1, \quad \forall, x = 1, \ldots, W; , y = 1, \ldots, H. \tag{1}$$

Variable sampling (Foveation). The sampling approach follows Poggio et al. [21] (see also [5, 24]), which modeled the human representation of visual information in the retina and the visual cortex. The *variable resolution* scheme (see SFigure 1) consists of sampling with a receptive field size that increases with eccentricity (distance from the fixation point). We apply a Log-Polar transformation to each image, where sample density remains constant $\forall\theta \in [0, 2\pi]$ and decreases linearly with r:

$$(r, \theta) = (\log(\sqrt{(x - x_f)^2 + (y - y_f)^2}), \arctan(\frac{y - y_f}{x - x_f})) \tag{2}$$

where x_f, y_f are the coordinates of the fixation point. The sampling scheme selects a fixed number of samples, N, distributed across the image. The sample positions are determined by defining concentric annuli around the fixation point, each containing a predefined number of samples proportional to its area. The area decreases linearly with r, which is akin to the increasing receptive field sizes in the human retina [5, 21, 24].

For the 3% sampling density, this results in roughly $N_{\text{var}} = 10K$ pixel samples falling within the FOV of an image with size 600×400 (quite typical for the COCO [18], GQA [12] etc.). We refer to this as a *3% sampling density*, equivalent to a 33x image down-scaling factor, with the number of samples increasing proportionally for smaller down-scaling factors.

Uniform sampling. Pixels are sampled uniformly across the image to achieve a total of N samples. The sampling map, S_{uni}, is generated by dividing the image into a log-polar grid, again using Equation 2, but here the concentric annuli maintain a constant area with the condition $N = N_{\text{var}}$. This creates even distribution of samples across the entire image while ensuring *information matching* between the images (see main paper "Information-matched images" section). The S_{uni} function is akin to simply down-scaling an image (losing information) and then up-scaling it back again using an interpolation algorithm (e.g., bilinear interpolation). Formally:

$$\sum_{x=1}^{W} \sum_{y=1}^{H} S_{\text{uni}}(x, y) = N_{\text{var}}. \tag{3}$$

1.1. Image reconstruction

For each sampling map $S \in \{S_{\text{var}}, S_{\text{uni}}\}$ and original image $I : W \times H \rightarrow \{0, ...256\}^3$, the sampled image is:

$$I_{\text{sampled}}(x, y) = S(x, y) \cdot I(x, y). \tag{4}$$

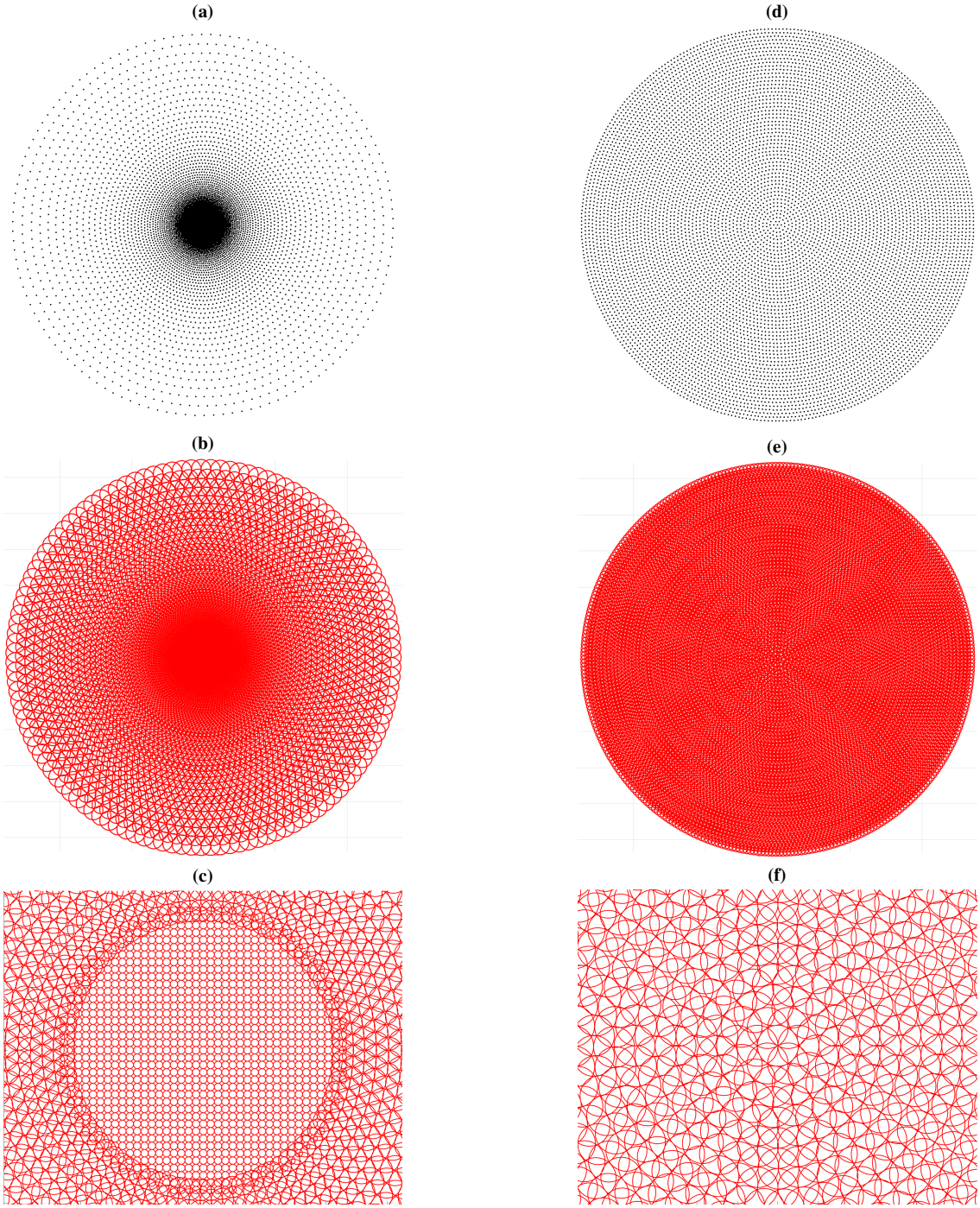
We reconstruct the full-resolution image $\hat{I}$ using an interpolation function $\mathcal{I}$ (e.g., bilinear interpolation):

$$\hat{I} = \mathcal{I}(I_{\text{sampled}}). \tag{5}$$

1.2. Code

We provide the code for generating the final images from our sampling maps in the Supplementary code on the website: https://seeingmorewithless.github.io/.

To understand the representational changes induced by variable sampling, it is essential to examine the underlying architectures of the models in question. We focus on two primary architectures from existing literature: DETR [4] and its extension, MDETR [14].



SFigure 1. **Sampling and interpolation schemes. (a,b,c)** Variable resolution sampling scheme with peak sample density at the center of fixation and linearly decreasing number of samples with eccentricity. The samples have non-overlapping receptive fields (spatial support) at the center of fixation (aka. 'foveola'), supporting a single pixel each. Outside the center, the receptive fields are overlapping and linearly increasing with eccentricity towards the periphery [5, 21, 24]. **(d,e,f)** Uniform resolution sampling scheme with a constant density of samples. Both schemes distribute an equal number of samples over the entire field of view (FOV) using log-polar coordinates. In this work, we interpolate filtered images from the sampled images, by applying a Gaussian filter around each sample with a standard deviation equal to the sample's receptive field radius.

2. Architectures

Studying the representations learned in large foundation models (such as BLIP2 [16], LLaVa [19], InstructBLIP [6]) in the context of our study requires *complete re-training from scratch* on sampling-scheme pre-processed versions of each dataset used in the models. Simply *fine-tuning* those models is not enough in our case, as we modify the input images fundamentally from the present paradigm. The base-level visual features extracted by training with uniform-resolution images, as is commonly done in the community and in CLIP [23] which the above models inherit from, would not be representative of the internal adaptations that occur during learning with variable sampled images.

Given that CLIP [23] was trained on 400 million image-text pairs for roughly 20 days on 592 V100 GPUs, the task of complete re-training has become largely inaccessible to the scientific community. As such, we decided to train from scratch only DETR (object detection model), and fine-tune MDETR among our VLMs. We study the impact of variable sampling on the learned representations of DETR and in this section, we establish its relation to MDETR. This is important context for how our study relates to the representations on the VQA task.

2.1. DETR Architecture

We begin by revisiting the architectures. DETR (**DE**tection **TR**ansformer) introduces a novel approach to object detection by leveraging transformer-based architectures. As depicted in SFigure 2, the model consists of three main components:

1. **Backbone**: A convolutional neural network (CNN) extracts feature maps from the input image.
2. **Transformer Encoder-Decoder**: The encoder processes the flattened feature maps with positional encodings, while the decoder generates a fixed set of object queries that attend to the encoder's output.
3. **Prediction Heads**: Each decoder output embedding is passed through a feed-forward network (FFN) to predict bounding box coordinates and class labels.

DETR employs a set-based bipartite matching loss to facilitate end-to-end training without the need for anchor boxes or non-maximum suppression, enabling direct prediction of object sets.

2.2. MDETR Architecture

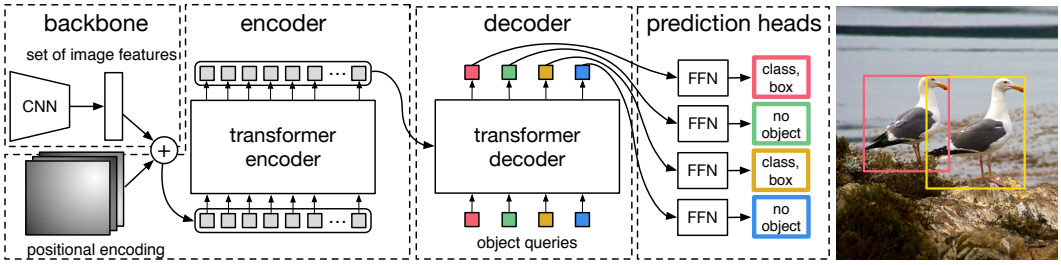
MDETR (**M**odular **DE**tection **TR**ansformer) [14] extends DETR to multimodal tasks by incorporating textual inputs, enabling the model to perform visual question answering (VQA) and referring expression comprehension (SFigure 3). The main components are:

1. **Multimodal Input Encoding.** MDETR encodes the image using a CNN backbone, similar to DETR, resulting in visual features. The text (e.g., a question or referring expression) is encoded using a pre-trained language model (e.g., RoBERTa) to obtain textual feature embeddings. Both visual and textual features are projected into a shared embedding space.
2. **Cross-Modal Transformer Encoder.** The projected visual and textual features are concatenated to form a single sequence and passed through a transformer encoder. This cross-modal encoder uses self-attention to model interactions between visual and textual tokens, allowing the model to align textual references with visual content.
3. **Transformer Decoder.** Similar to DETR, MDETR uses a transformer decoder with object queries. The decoder cross-attends to the outputs of the cross-modal encoder, enabling it to integrate information from both modalities. The decoder outputs are used to predict bounding boxes corresponding to objects referred to in the text.
4. **Output Heads and Loss Functions.** In addition to bounding box regression and classification, MDETR employs specialized loss functions for multimodal alignment, such as the soft token prediction loss and contrastive alignment loss.

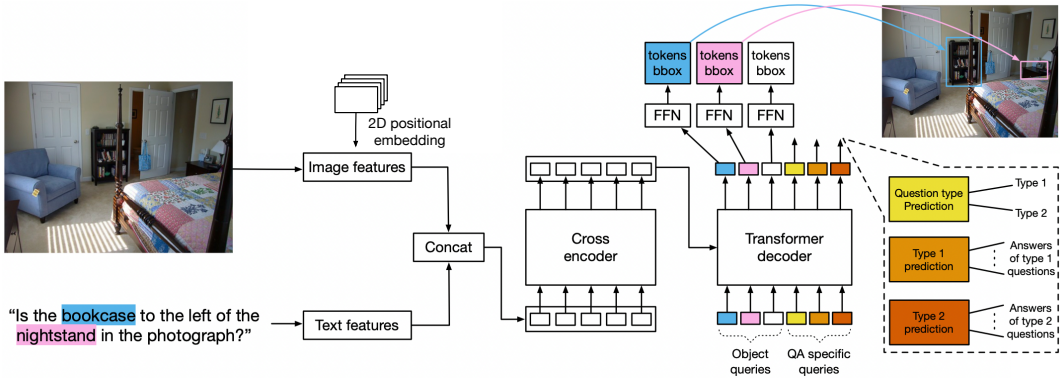
2.3. Representational similarity

Despite being designed for different tasks, DETR and MDETR share a common architectural foundation centered around transformer-based encoder-decoder mechanisms that process visual features extracted by a CNN backbone. Both models process visual features through an encoder, which is the main component of our analysis. In our setup, both models also use the same CNN backbone (ResNet101 [10]), trained from scratch, on which we also conduct the "Neuronal specialization" experiment from the main paper (Section "Human-like representations", point II.).

By training the DETR model on the semantically rich task of object detection (involving segmentation masks, captions, locations, bounding boxes etc.) we aim to address from the perspective of vision what adaptations occur under variable sampling during complex visual tasks. We refer the reader to the main paper for our analysis, Section "Human-like representations".



SFigure 2. **DETR architecture.** Illustration adapted from [4]. DETR architecture consisting of; (1) CNN-backbone; (2) transformer encoder-decoder; (3) feed-forward network (FFN).



SFigure 3. **MDETR architecture.** Illustration adapted from [14]. The architecture consists of: (1) a CNN backbone for visual feature extraction and a pre-trained language encoder; (2)(3) a transformer encoder-decoder for cross-modal interaction and object query processing; and (4) output heads for multimodal alignment (FFN).

3. VQA

Here we provide additional results on the effects of fixation point location. We observe that shifts cause negligible differences in performance (STable 1).

3.1. Qualitative evaluation

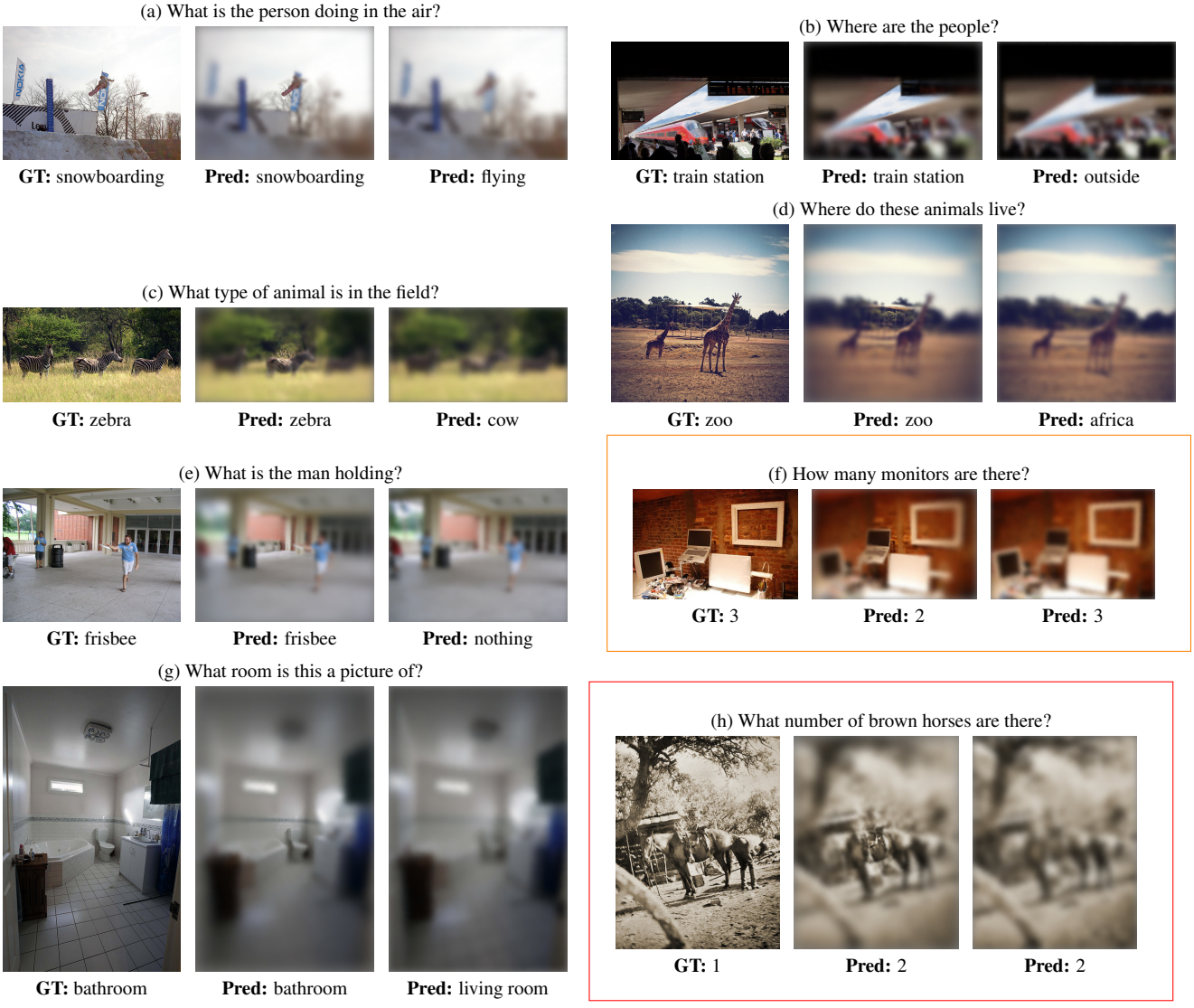
While our evaluation focuses on quantitative analysis, it is important to recognize the limitations of only using summary statistics. A major concern is the validity of benchmarks [17] — it is always questionable if a benchmark really measures what it claims to, especially as we find "shortcuts" that machine learning models tend to use [9, 22]. In effort to partially address this issue, we provide a qualitative evaluation of several models on the VQA task — showing examples both in favor of variable and in favor of uniform sampling (SFigure 4).

3.2. Detailed VQA results

We provide detailed results on the VQA task for all our models at the 3% density (STable 2, 3, 4, 5, 6).

STable 1. Performance accuracy comparison between a variable central fixation (column "Variable") and corner fixations across different models and datasets, compared to uniform sampling (column "Uniform"). Results at 3% density. We define a corner fixation as in the main paper, 100 pixels away from the center along the given diagonal (TR: top-right, TL: top-left, BR: bottom-right, BL: bottom-left). We report the standard deviation of accuracy across the center and the different fixations (column "Variable Std.").

Model	#Total Params	Dataset	Variable	Variable Std.	Uniform
MDETR-ResNet101-RoBERTa [14]	169M	GQA [12]	46.79%	±0.01%	44.13%
BLIP-2-FlanT5$_{XL}$ [16]	3.4B	GQA	42.27%	±0.21%	40.72%
BLIP-2-FlanT5$_{XL}$	3.4B	VQAv2 [8]	57.89%	±0.46%	56.19%
InstructBLIP-FlanT5$_{XL}$ [6]	4B	VQAv2	66.37%	±0.56%	66.48%
ViLT-B/32 [15]	87.4M	VQAv2	64.90%	±0.82%	63.01%
LLaVa-v1.5 [19]	13B	VQAv2	65.91%	±0.75%	65.14%



SFigure 4. **VQA examples** from LLaVa-v1.5 [19] and ViLT [15] on VQAv2 [8] at 3% density. Variable (mid), Uniform (right). **(a,b,c,d,e,g)** Examples of fine details (imperfectly placed, **b**) yielding correct results. **(f)** Example where the variable scheme struggles with *very* peripheral objects (main paper, Section 3.2(1)). **(h)** Where both schemes fail due to insufficient resolution at critical regions.

STable 2. **ViLT [15] on VQAv2[8]**

Model	Question Type	Baseline	Variable	Uniform
Pretrained	Other	75.57%	**56.51%**	54.16%
	Number	64.10%	**43.62%**	41.76%
	Yes/No	95.74%	**83.42%**	82.05%
	Total	81.64%	**64.93%**	63.01%

STable 3. **BLIP2 [16] on VQAv2 [8]**

Model	Question Type	Baseline	Variable	Uniform
Pretrained	Yes/No	84.27%	**80.46%**	79.69%
	Number	40.97%	**36.02%**	33.81%
	Other	52.90%	**46.50%**	44.23%
	Total	63.12%	**57.89%**	56.19%

STable 4. **LLaVa [19] on VQAv2 [8]**

Model	Question Type	Baseline	Variable	Uniform
Pretrained	Yes/No	92.81%	85.15%	**84.66%**
	Number	61.32%	44.52%	**43.92%**
	Other	71.71%	**56.95%**	55.91%
	Total	78.27%	**65.91%**	65.14%

STable 5. **InstructBLIP [6] on VQAv2 [8]**

Model	Question Type	Baseline	Variable	Uniform
Pretrained	Yes/No	90.14%	85.41%	**85.60%**
	Number	54.62%	46.16%	**47.06%**
	Other	65.82%	**57.24%**	57.07%
	Total	73.49%	66.37%	**66.48%**

STable 6. **MDETR [14] model evaluation on GQA-testdev [12]**

Model	Question Type	Baseline Accuracy	Variable Accuracy	Uniform Accuracy
Pretrained	(1) Object existence	95.6%	**90.1%**	89.9%
	(2) Object attribute	71.2%	**58.2%**	57.6%
	(3) Object category	76.0%	**65.8%**	61.3%
	(4) Object relation	53.1%	**37.5%**	34.2%
	(5) Global scene	95.8%	**94.2%**	93.3%
	Total	61.7%	**46.8%**	44.1%
Fine-tuned	(1) Object existence	95.6%	**93.8%**	93.4%
	(2) Object attribute	71.2%	**62.6%**	62.6%
	(3) Object category	76.0%	**71.1%**	70.3%
	(4) Object relation	53.1%	**46.5%**	44.9%
	(5) Global scene	95.8%	95.2%	**95.7%**
	Total	61.7%	**54.3%**	53.3%

4. Object detection

STable 7 was referred to in the main paper in the **Counting samples (2)** experiment. It shows the performance of object detection models when evaluated to detect objects encompassed by an *equal number* of samples (see Figure 1a,d). That is, a subset of COCO containing *only information-matched objects* (as computed by the segmentation mask overlaid onto the sampling map) was used in this evaluation. This subset contains mostly peripheral objects, since in order to achieve an equality of samples with the uniform scheme, the objects should clearly fall quite far from the center of the fixation point (which has a very high sampling density). This is, in a sense, the most "sterile" setting for studying the effects of variable sampling; it fully excludes the effects of language, fixation point location, etc.

STable 7. **Sample-equalised evaluation.** Segmentation Average Recall (AR, IoU=0.50:0.95) measured only for objects covering an equal number of samples with both variable and uniform sampling schemes. The subscripts $_{S}$, $_{M}$, $_{L}$ correspond to performance on small, medium, and larger objects.

Model	Sampling	AR	AR_S	AR_M	AR_L
DETR-R101	Baseline	54.8	15.3	51.9	72.0
DETR-R101	Uniform	37.1	1.2	28.0	51.6
DETR-R101	**Variable**	**38.5**	**2.1**	**31.0**	**54.7**
Mask RCNN-R101	Baseline	52.5	25.6	49.7	64.2
Mask RCNN-R101	Uniform	34.7	1.6	27.7	48.2
Mask RCNN-R101	**Variable**	**36.9**	**3.4**	**31.7**	**48.6**

SFigure 5 qualitatively compares detection results of the DETR object detector between the variable and uniform sampling schemes, including object category, bounding box and instance segmentation.

SFigure 6 shows additional detection results of the DETR object detector for input images at baseline, uniform and variable resolutions. Self-attention maps at four locations around the boundaries of each image, show the different attention patterns in the model for each sampling scheme (see main paper Section "Human-like representations" (I) for an explanation of those maps).

5. Human-like representations

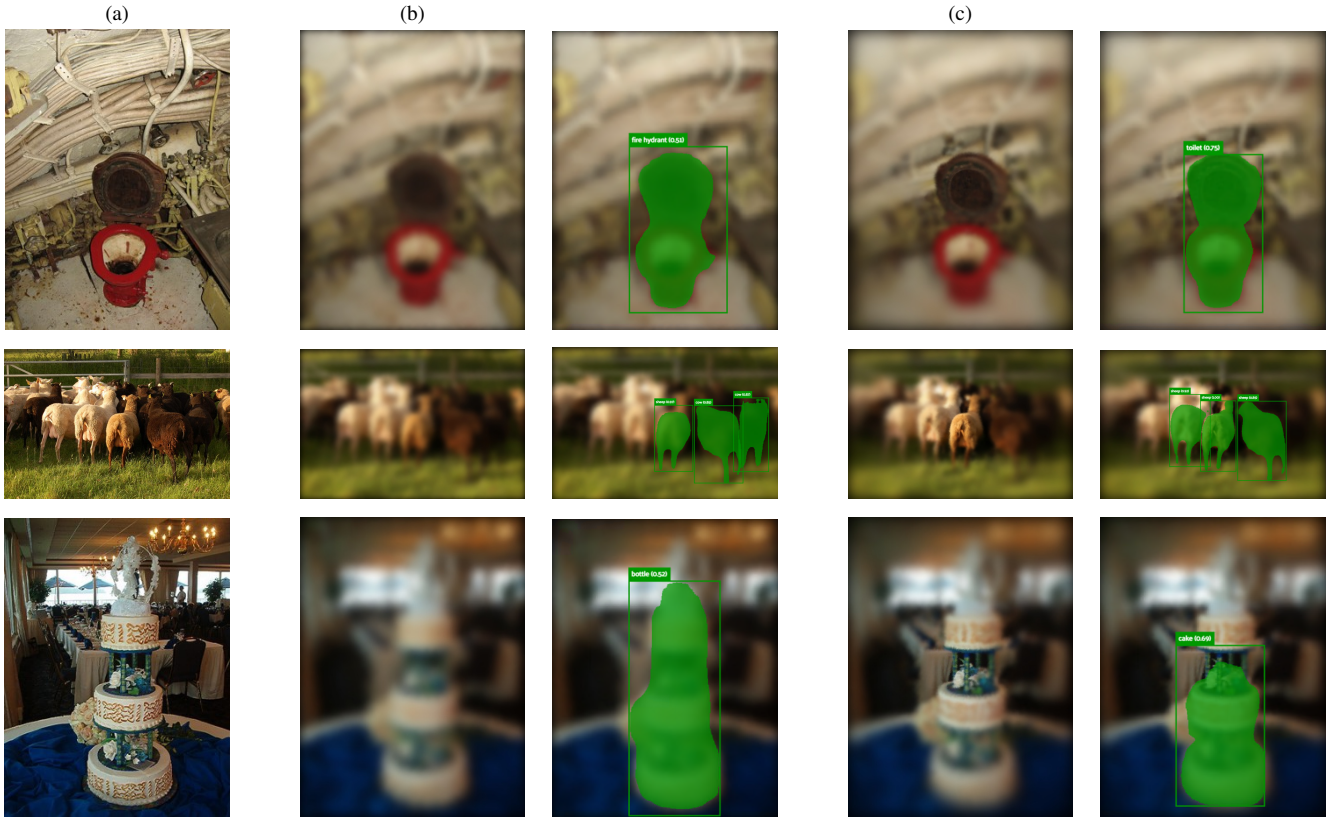
II. Single model generalizes to detect multi-resolution spanning objects by allowing resolution-specialization in CNNs. In the main paper, we hypothesized that a network trained with variable resolution input will keep some separation between the neurons it dedicates to specific resolutions, even for single object instances. Whether this occurs is not a trivial question, since it is entirely possible that the neurons of the network have learned the average resolution in our training set only: producing high dot-products (activations) for mid-resolution occupying object segments and low everywhere else.

Here we provide details on the experimental setup around this claim. To test our hypothesis, we fed the variable-resolution trained ResNet101 [10] (backbone of MDETR, DETR, and MaskRCNN [4, 11, 14]) images of Type 0 and Type 1 resolution (SFigure 7b,7c). Both of those resolutions are from the spectrum contained in the original training images, but the latter (Type 0 resolution) has its fixation shifted away from the center. As such, both images represent the same central crop, but in low variable resolution (Type 0), and high variable resolution (Type 1). We inferred 2,500 Type 0 and 2,500 Type 1 images (splitting the COCO validation set in two) to MaskRCNN. We extracted the feature map tensors produced by a deep layer in the ResNet101 backbone ($backbone.body.layer3.block22.batchnorm3$), which resulted in two sets of 2,500 tensors, with size varying from $1024 \times 50 \times 50$ to $1024 \times 70 \times 70$, depending on the size of the inferred image. For each 2D sub-matrix in the tensor (1,024 sub-matrices in total), we took the *average, median and maximum* of the neuronal (kernel) activation. Those metrics serve as *descriptive statistics* for our following experimentation and reduce our datasets to *two sets* of $2,500$ tensors, each of which has dimension $1024 \times 3 \times 1$.

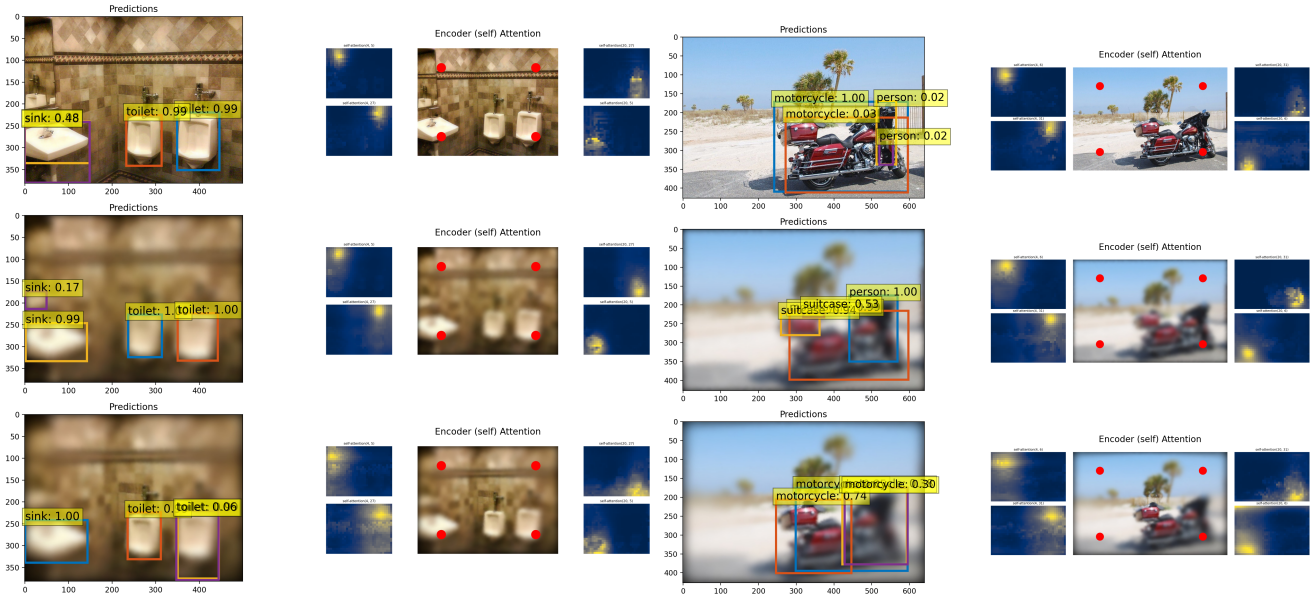
Given a sample of total $5,000$ tensors (each with size $1024 \times 3 \times 1$), generated by the two different types of source images (*Type 1* and *Type 0* resolution), we want to establish whether there's a statistically significant difference between the activation pattern of the neurons *solely* depending on the resolution type. In the setting of this experiment, we have ensured that all other factors have been kept the same in the generation of the tensors (same object distribution between the groups etc). Formally, let $X_1, \ldots, X_{2,500}$, $Y_1, \ldots, Y_{2,500}$ be independent, random vectors of real numbers, representing our tensors. We have $X_i \sim \mathbb{P}_0$ and $Y_i \sim \mathbb{P}_1$, with $\mathbb{P}_0$, $\mathbb{P}_1$ probability distributions. The hypothesis test is then reduced to:

$$H_0 : \mathbb{P}_0 = \mathbb{P}_1 \text{ vs. } H_1 : \mathbb{P}_0 \neq \mathbb{P}_1 \tag{6}$$

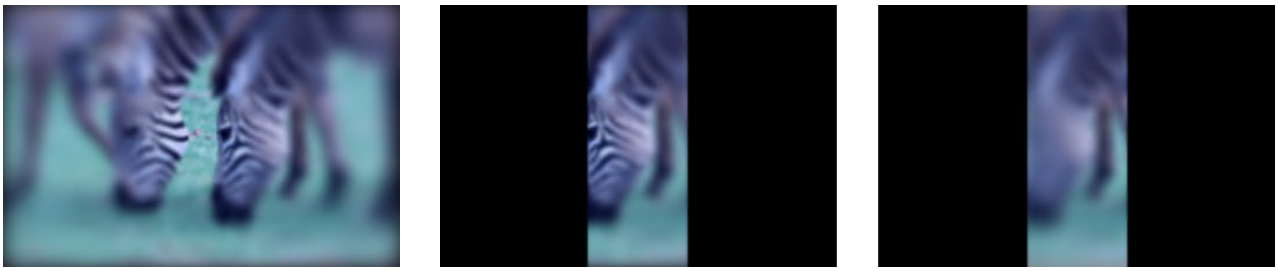
Most two-sample tests assume some extent of normality. Lacking further information on the inner machinery of the training process and the distribution of kernel populations, we must establish a non-parametric, general approach where minimal assumptions are made about the tensor generation process. Many non-parametric models, however, such as kernel density estimation, are not typically feasible



SFigure 5. **Detection examples of the DETR model.** We see in several instances that the variable resolution model benefits from the texture of an object, critical for determining its class correctly. (a) Full-resolution image. (b) Uniform sampling scheme. (c) Variable sampling scheme.



SFigure 6. **Additional interpretability and detection examples in DETR model.** Top to bottom: Baseline (full), uniform and variable resolution.



(a) CNN training image (b) Type 1 resolution (c) Type 0 resolution

SFigure 7. **Shifted fixation points.** Example of cropped images around the center, with center fixation and shifted fixation.

in high-dimensional settings [1]. Consequently, we need a model that doesn't require an intermediate density estimation to avoid the curse of dimensionality. We use a classification model as a proxy for a two-sample test.

Classification. Let $\mathbb{P}_0, \mathbb{P}_1$ be two distinct probability distributions. Let $\mathcal{C}$ a classifier model. $\mathcal{C}$ is a simple function from the set of all possible tensor samples taken from $\mathbb{P}_0$, $\mathbb{P}_1$ to {0,1}, indicating whether the sample belongs to $\mathbb{P}_0$ or $\mathbb{P}_1$. We can denote the function as $\mathcal{C} : \mathbb{X} \to \{0, 1\}$, representing a logistic-regression classifier (fit on at most $1,024 \times 3 \times 1 = 3,072$ variables, not considering regularization).

Two-sample test based on classification accuracy. Classification-based approaches have been proposed for two-sample testing on high-dimensional, complex data. [2, 3, 7, 13, 20]. If a classification model can sufficiently discriminate between samples from two populations, it's reasonable to assume that the underlying data generation processes are different. Consequently, the accuracy of a classification model (preferably, if not imperatively, over out-of-sample data) must be a constituent of the test statistic. [13, 20] discuss the consistency and power of a classifier-based test compared to the minimax power, [3] proposes an estimation of the likelihood ratio by the odds ratio of the classification probability, and [7] uses the score assigned to each datapoint by the classifier to reduce the dimensions and run a single-dimensional two-sample test.

Notation. We first introduce the notation necessary to study the classifier's accuracy. Let $\mathcal{X} = \{X_i\}_{i=1}^{n_0}$ and $\mathcal{Y} = \{Y_i\}_{i=1}^{n_1}$ be independent random vectors representing our tensors, drawn from probability distributions $\mathbb{P}_0$ and $\mathbb{P}_1$, respectively. These correspond to *Type 0* and *Type 1* tensors. We split the data into training and test sets: $\mathcal{X}_{\text{train}}$, $\mathcal{Y}_{\text{train}}$ for training, and $\mathcal{X}_{\text{test}}$, $\mathcal{Y}_{\text{test}}$ for testing, with sizes tr_0, tr_1, te_0, and te_1, respectively, such that $n_0 = tr_0 + te_0$ and $n_1 = tr_1 + te_1$. $\mathcal{X}$ and $\mathcal{Y}$ contain Type 0 and Type 1 tensors, respectively (each tensor X_i, Y_i of size $1024 \times 3 \times 1$).

We train a classifier $\mathcal{C}$ on the training samples $\mathcal{X}_{\text{train}}$ and $\mathcal{Y}_{\text{train}}$. We then evaluate the classifier on the test samples $\mathcal{X}_{\text{test}}$ and $\mathcal{Y}_{\text{test}}$ to estimate the accuracies:

$$\hat{\alpha}_0 \equiv \frac{1}{te_0} \sum_{i=1}^{te_0} \mathbb{1}\left(\mathcal{C}(X_i) = 0\right), \quad \hat{\alpha}_1 \equiv \frac{1}{te_1} \sum_{i=1}^{te_1} \mathbb{1}\left(\mathcal{C}(Y_i) = 1\right). \tag{7}$$

Here, $\hat{\alpha}_0$ is the proportion of correctly classified samples from $\mathbb{P}_0$, and $\hat{\alpha}_1$ is the proportion of correctly classified samples from $\mathbb{P}_1$. These sample proportions are unbiased estimators of the expected accuracies α_0 and α_1, defined as:

$$\alpha_0 = \mathbb{E}_{X \sim \mathbb{P}_0}[\mathbb{1}(\mathcal{C}(X) = 0)], \quad \alpha_1 = \mathbb{E}_{Y \sim \mathbb{P}_1}[\mathbb{1}(\mathcal{C}(Y) = 1)]. \tag{8}$$

Under the null hypothesis $H_0 : \mathbb{P}_0 = \mathbb{P}_1$, and assuming no predictive power, we have $\alpha_0 = \alpha_1 = 0.5$. As the sample sizes te_0 and te_1 increase, the estimators $\hat{\alpha}_0$ and $\hat{\alpha}_1$ converge to their expected values α_0 and α_1 by the law of large numbers. The overall test accuracy is then:

$$\hat{\alpha}_* \equiv \frac{te_0\hat{\alpha}_0 + te_1\hat{\alpha}_1}{te}, \quad \text{where} \quad te = te_0 + te_1. \tag{9}$$

Similarly, $\hat{\alpha}_*$ is an estimator of the overall expected accuracy α_*, defined as:

$$\alpha_* = \frac{te_0\alpha_0 + te_1\alpha_1}{te}. \tag{10}$$

Intuitively, if the null hypothesis H_0 holds, the classifier cannot distinguish between the two distributions, and thus $\alpha_0 = \alpha_1 = 0.5$, leading to α_* converging to 0.5.

Half-permutation method. With this notation in mind, we follow Ilmun Kim [13] and divide the data accordingly. After training the logistic regression classifier $\mathcal{C}$ on $\mathcal{X}_{\text{train}}$ and $\mathcal{Y}_{\text{train}}$, we measure its accuracy $\hat{\alpha}_*$ over the test samples $\mathcal{X}_{\text{test}}$ and $\mathcal{Y}_{\text{test}}$.

We then merge $\mathcal{X}_{\text{test}}$ and $\mathcal{Y}_{\text{test}}$ into one set $\mathcal{Z}$, permute the labels randomly, and split them into two equal halves to generate new datasets $\mathcal{Z}_1$ and $\mathcal{Z}_2$. By testing the trained classifier $\mathcal{C}$ on these permuted datasets, we obtain permuted

accuracies $\alpha_1, \alpha_2, \ldots, \alpha_P$. Under the null hypothesis, these permuted accuracies are estimators of the expected accuracy when the labels are independent of the inputs. As P increases, they provide an accurate estimate of the null distribution of the accuracy estimator. We repeat this process P times, choosing $P > (1-\alpha)/\alpha$ (e.g., $P \geq 100$ for $\alpha = 0.01$).

Sorting $\hat{\alpha}_*, \alpha_1, \ldots, \alpha_P$, we assign an order to each accuracy. The hypothesis test can be represented by the indicator function:

$$\mathcal{H}_p \equiv \mathbb{1}\left(\hat{\alpha}_* > \alpha^{(k)}\right), \tag{11}$$

where $\alpha^{(k)}$ is the kth order statistic of the sample, with $k = \lceil (1-\alpha)(1+P) \rceil$. Essentially, if the classifier's observed accuracy $\hat{\alpha}_*$ significantly exceeds the accuracies obtained due to "luck" (i.e. the most predictive random label assignment), it suggests that the classifier captures true differences between $\mathbb{P}_0$ and $\mathbb{P}_1$. [13] proves the consistency of this test and that the size, asymptotically and under the null hypothesis, will be less than α.

Setup. The authors suggest using $n \geq 400$ samples for high-dimensional data ($d \geq 200$) and a train-test ratio κ of $1/2$. We used $n = 5000$ total samples, with $\kappa = 1/2$, resulting in $tr_0 = tr_1 = 2500$ training samples and $te_0 = te_1 = 2500$ test samples from each distribution.

Restricting $P > (1-\alpha)/\alpha$ for $\alpha = 0.01$, we have $P \geq 100$. We set $P = 1000$, also testing $P = 100$, and achieved similar results. The classifier achieved an observed accuracy of $\hat{\alpha}_* = 95.32\%$. The permuted accuracies α_i ranged between 47.48% and 52.84%, centered around 50%, as expected under the alternative. Below are the sorted permuted accuracies and the true model accuracy $\hat{\alpha}_*$:

$$0.4748, 0.4804, \ldots, \quad 0.5196, 0.5228, 0.5284, \quad \textbf{0.9532}. \tag{12}$$

Conclusion. The test easily rejected the null hypothesis at $\alpha = 0.01$, indicating that the two tensors were drawn from *different* distributions. We conclude that there indeed are significant differences in the activation patterns of the neurons in the network governed *solely* by resolution. This suggests that the network indeed learns separate convolutional kernels for processing objects at different resolutions, rather than a single set of uniformly applicable kernels with less sensitivity for any given resolution type. Figure 6g,h in the main paper shows the values of the most significant predictors learned by the Type 0/Type 1 logistic classifier. We can clearly see that the higher variable resolution images (Type 1) tend to induce a higher activation value in some neurons (**h**), while other neurons seem to fire *equally* for both resolution types (**g**). This indeed suggests some resolution specialization in the neurons of the network.

References

[1] Yasemin Altun, Karsten M. Borgwardt, Tim Dwyer, Peter Eades, Kenji Fukumizu, Evian Gordon, Arthur Gretton, Seok-Hee Hong, Bernhard Sch ölkopf, Alex Smola, Vishy Vishwanasan, Adel Ahmed, Kelvin Cheng, David Cho Yau Fung, Damian Merrick, Kathryn E. Merrick, Collin D Murray, Xiaobin Shen, Ying Xin Wu, and Ben Yip. Learning via hilbert space embedding of distributions. 2007. 9

[2] Gilles Blanchard, Gyemin Lee, and Clayton Scott. Semi-supervised novelty detection. *Journal of Machine Learning Research*, 11:2973–3009, 2010. 9

[3] Haiyan Cai, Bryan Goggin, and Qingtang Jiang. Two-sample test based on classification probability. *Statistical Analysis and Data Mining: The ASA Data Science Journal*, 13, 2019. 9

[4] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European conference on computer vision*, pages 213–229, 2020. 1, 4, 7

[5] Christine A Curcio, Kenneth R Sloan, Robert E Kalina, and Anita E Hendrickson. Human photoreceptor topography. *Journal of comparative neurology*, 292(4):497–523, 1990. 1, 2

[6] Wenliang Dai, Junnan Li, DONGXU LI, Anthony Tiong, Junqi Zhao, Weisheng Wang, Boyang Li, Pascale N Fung, and Steven Hoi. Instructblip: Towards general-purpose vision-language models with instruction tuning. In *Advances in Neural Information Processing Systems*, pages 49250–49267. Curran Associates, Inc., 2023. 3, 5, 6

[7] Jerome H. Friedman. On multivariate goodness of fit and two sample testing. *eConf*, C030908:THPD002, 2003. 9

[8] Yash Goyal, Tejas Khot, Aishwarya Agrawal, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. Making the V in VQA Matter: Elevating the Role of Image Understanding in Visual Question Answering. *International Journal of Computer Vision*, 127(4):398–414, 2019. 5, 6

[9] Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel Bowman, and Noah A. Smith. Annotation artifacts in natural language inference data. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 107–112, New Orleans, Louisiana, 2018. Association for Computational Linguistics. 4

[10] Kaiming He, X. Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2015. 3, 7

[11] Kaiming He, Georgia Gkioxari, Piotr Dollar, and Ross Girshick. Mask R-CNN. *Proceedings of the IEEE International Conference on Computer Vision*, pages 2980–2988, 2017. 7

[12] Drew A Hudson and Christopher D Manning. Gqa: A new dataset for real-world visual reasoning and compositional question answering. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6700–6709, 2019. 1, 5, 6

[13] Aarti Singh Larry Wasserman Ilmun Kim, Aaditya Ramdas. Classification accuracy as a proxy for two sample testing. *Annals of Statistics*, pages 411–434, 2020. 9, 10

[14] Aishwarya Kamath, Mannat Singh, Yann LeCun, Gabriel Synnaeve, Ishan Misra, and Nicolas Carion. Mdetr - modulated detection for end-to-end multi-modal understanding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1780–1790, 2021. 1, 3, 4, 5, 6, 7

[15] Wonjae Kim, Bokyung Son, and Ildoo Kim. Vilt: Vision-and-language transformer without convolution or region supervision. In *International Conference on Machine Learning*, pages 5583–5594, 2021. 5, 6

[16] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. BLIP-2: bootstrapping language-image pre-training with frozen image encoders and large language models. In *ICML*, 2023. 3, 5, 6

[17] Thomas Liao, Rohan Taori, Inioluwa Deborah Raji, and Ludwig Schmidt. Are we learning yet? a meta review of evaluation failures across machine learning. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021. 4

[18] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, pages 740–755. Springer, 2014. 1

[19] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. In *NeurIPS*, 2023. 3, 5, 6

[20] David Lopez-Paz and Maxime Oquab. Revisiting classifier two-sample tests. *ICLR 2017*, 2017. 9

[21] Tomaso Poggio, Jim Mutch, and Leyla Isik. Computational role of eccentricity dependent cortical magnification. Technical Report CBMM Memo 017, 2014. 1, 2

[22] Adam Poliak, Jason Naradowsky, Aparajita Haldar, Rachel Rudinger, and Benjamin Van Durme. Hypothesis only baselines in natural language inference. In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics*, pages 180–191, New Orleans, Louisiana, 2018. Association for Computational Linguistics. 4

[23] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763, 2021. 3

[24] Andrew B Watson. A formula for human retinal ganglion cell receptive field density as a function of visual field location. *Journal of vision*, 14(7):15–15, 2014. 1, 2