Appendix

A. Object Decoder Weight Initialization

Various state-of-the-art transformer models like BERT, OpenCLIP and GPT use heuristics for weight initialization, for example by initialising weights to a zero-mean normal distribution of standard deviation 0.02 or $\frac{1}{\sqrt{H}}$, for H the hidden dimension. Guided by the principles of mean field theory, we rigorously calculate weight initialization standard deviations such that the expected forward pass norms are statistically preserved throughout the object decoder, and such that the overall contributions of the main and residual pathways are balanced, avoiding excessive input signal dilution. This comes with numerical and gradient stability benefits, leading to fast and robust initial convergence.

A.1. Strategies for Strong Initial Convergence

In general, the rule-of-thumb aims of the presented *variance-balanced weight initialization strategy* for the forward pass of a network are:

- **Avoid input signal dilution.** For example, if the residual pathway of the input through the network involves repeatedly combining it with values calculated from random weights and renormalizing, then the 'proportion' of the final output signal that actually corresponds statistically to the input signal can be exponentially microscopic, making initial convergence more difficult.
- **Avoid combining unbalanced scales of data.** If two signals are added or concatenated that have significantly different scales (multiplication is less critical), especially if done repeatedly, then this has potentially negative numerical effects on backpropagation. It can also quickly lead to signal dilution.
- **Avoid significant scale decay/growth.** For example, if each residual block in a network initially causes a rescaling of signals by only 50% on average, then a moderate 12-layer transformer with cross-attention will initially produce outputs that are only one 70 billionth of the inputs, which is undesirable for learning.
- **Aim for around unit scale throughout.** Making an optimizer deal with vastly different scales of signals and gradients throughout the network is undesirable, even if the optimizer has features to help deal with it. Floating-point representations have their limits, especially if attempting to train in reduced or mixed precision. Certain layers, like for example certain activation functions (*e.g.* GELU), are also 'designed' for approximately unit standard deviation scales of input data.

Based on these rules of thumb, it can immediately be seen why transformers with post-layer normalization [68] in general struggle to train for larger numbers of layers. In each residual block, the initially random weights in the residual path cause an essentially random residual signal to be added to the input signal, before the combined signal is then layer-normalized. This results in the input signal to the next residual block only being a certain fraction the input signal, and the rest just random noise. As this combined signal is now the new input signal for the next block, it further downscales the original input signal by a certain factor and adds even more random noise. This process continues throughout the entire transformer, leading to prototypical signal dilution, with the problem getting *exponentially* worse for larger numbers of layers. This is why pre-layer normalization was chosen for NOVIC.

A.2. Data Variance Propagation

We must first understand how variances propagate through the individual layers of a transformer network before we can integrate this knowledge into a complete strategy.

Addition. If a vector containing elements of variance σ_1^2 is added to an (uncorrelated) vector of variance σ_2^2, then the variance of the elements in the sum of the two vectors is

$$\sigma_{1+2}^2 = \sigma_1^2 + \sigma_2^2. \tag{1}$$

Linear layer. If a vector $\mathbf{v} = (v_1, \ldots, v_D)$ of dimension D containing elements of variance σ_D^2 is passed to a (zero-bias or biasless) linear layer of output dimension T, then given weights w_{ij} of variance σ_w^2, the expected variance of the output elements $\mathbf{u} = (u_1, \ldots, u_T)$ is

$$\begin{aligned}\sigma_T^2 &= \text{Var}(u_i) = \text{Var}\left(\sum_{j=1}^{D} w_{ij}v_j\right) = \sum_{i=1}^{D} \text{Var}(w_{ij}v_j) \\ &= \sum_{i=1}^{D} \text{Var}(w_{ij}) \cdot \text{Var}(v_j) = D \cdot \sigma_w^2 \cdot \sigma_D^2. \end{aligned} \tag{2}$$

A.3. Variance-Balanced Transformer Initialization

In terms of the layers that are relevant for the statistical analysis of the data passing through the model during training, the object decoder fundamentally consists of (cf. Fig. 3):

- **Input stage:** A linear projection of the input embedding vector in parallel to the token embeddings of the ground truth object noun tokens, followed by an additive positional embedding,
- **Transformer:** L consecutive transformer decoder layers, each sequentially consisting of two residual blocks, first for multi-head attention and then for feedforward layers, and,

- **Output stage:** A layer normalization, followed by a linear projection to obtain the token logits, where the weights of the projection are tied to the token embeddings matrix.

Note that no biases are used throughout the architecture, including in all linear, layer norm, and multi-head attention layers, and thus can be mathematically neglected in the analysis. If they were present, they would be initialized to zero anyway however, to allow them to be used in the long run by the optimizer, but initially not affect the optimization too much. We proceed with details of the variance-balanced weight initialization strategy for each part of the transformer model.

Input stage. The input to NOVIC is a unit embedding vector $\mathbf{e} = (e_1, \ldots, e_F)$ of dimension F (*e.g.* $F = 768$ for the nominal CLIP embedder), so the elements of this input signal vector have a zero-mean variance of

$$\sigma_e^2 = \frac{1}{F}\sum_{i=1}^{F} e_i^2 = \frac{1}{F}\|\mathbf{e}\|^2 = \frac{1}{F}. \tag{3}$$

To ensure approximate unit scale throughout the transformer post-initialization, we wish for each signal element throughout the network to have near-unit variance. As such, we initialize the weights W_p of the linear projection layer from a zero-mean normal distribution of variance $\frac{1}{2}$, *i.e.* $\mathcal{N}(0, \frac{1}{2})$, and do the same for all weights in the token embeddings W_{te} and learned position embeddings W_{pe}. If W_{te}^* is the learnable token embeddings for the specific ground truth tokens being passed to the object decoder, then from Eqs. (1) to (3), we can see that the expected variance σ_p^2 post-positional embedding, *i.e.* of the actual input sequence vectors to the transformer layers, is thus

$$\begin{aligned}
\sigma_p^2 &= \mathrm{Var}\Big(\mathrm{Concat}(\mathbf{e}W_p, W_{te}^*) + W_{pe}\Big) \\
&= \mathrm{Concat}\Big(\mathrm{Var}(\mathbf{e}W_p), \mathrm{Var}(W_{te}^*)\Big) + \mathrm{Var}(W_{pe}) \\
&= \mathrm{Concat}\Big(F \cdot \mathrm{Var}(W_p) \cdot \sigma_e^2, \tfrac{1}{2}\Big) + \tfrac{1}{2} \\
&= \mathrm{Concat}\Big(F \cdot \tfrac{1}{2} \cdot \tfrac{1}{F}, \tfrac{1}{2}\Big) + \tfrac{1}{2} \\
&= 1,
\end{aligned} \tag{4}$$

as desired. Note that some simplifying leniency is being used in the notation here by referring to elementwise variances with $\mathrm{Var}(\cdot)$ of a matrix, and to work around the detail that the PH outputs of the linear projection layer need to be split into P (number of prefix tokens) individual sequence vectors of dimension H for the dimensions to work out.

Transformer. The decoder-only transformer used in NOVIC consists of L layers, each containing two residual blocks, for a total of $2L$ consecutive residual blocks that strictly add calculated residuals to the input signal but otherwise leave it untouched. Due to the random weights used in the calculation of the residual contributions, the residual blocks can be modeled as adding statistically independent noise to the main path that we want to ensure does not excessively dilute the input signal. We initialize the weights of each residual pathway so that it contributes an expected variance of $\frac{1}{2L}$, leading to a final expected output variance of 2, of which half statistically comes directly from the input signal. The immediately following layer normalization of the output stage then renormalizes the scale of the data.

We achieve a residual pathway variance of $\frac{1}{2L}$ in each self-attention block by using weights of all 1 for the leading layer normalization, using weights of variance $\frac{1}{H}$ in order to make the query-key-value projections variance-preserving, and using weights of variance $\frac{1}{2LHS}$ in order to generate the desired output variance while accounting for (using S) the variance reduction that occurs in the scaled dot-product attention (SDPA). It can be shown that for P prefix tokens and an expected input variance σ^2 to the SDPA, a good approximation of the variance reduction is given by the factor

$$S = \frac{1}{P}\left(1 + \sigma^4 \frac{(P-1)}{P}\right). \tag{5}$$

As the layer normalization layer outputs unit variance signals and the query-key-value projections are variance-preserving, the expected input variance to the SDPA is $\sigma^2 = 1$, so this reduces to

$$S = \frac{2P-1}{P^2}. \tag{6}$$

For the residual pathway of each feedforward block we use a similar strategy, and use weights of all 1 for the leading layer normalization, use weights of variance $\frac{1}{H}$ in order to make the first feedforward linear layer variance-preserving (refer to Eq. (2)), and use weights of variance $\frac{1}{2LKA}$ for the second feedforward linear layer, where K is the intermediate feedforward dimension, and the factor A compensates for the variance reduction of the activation function. A is estimated by transforming the expected unit normal probability distribution by the activation function and calculating the variance around zero of the resulting distribution. For ReLU this yields $A = \frac{1}{2}$, and for GELU this yields $A \approx 0.4252$.

Output stage. As the weights of the token logits linear layer is tied to the already-initialized learnable token embeddings (variance $\frac{1}{2}$), it only remains to initialize the weight of the final layer normalization that precedes it (required due to the use of pre-layer normalization). As the final token logits need to be a suitable scale for the log-softmax operation that follows as part of the loss, we effectively 'normalize' the output sequence vectors by initializing the output layer normalization weights to $\sigma_o = \frac{1}{\sqrt{H}}$. Using Eq. (2), this results in token logits of standard deviation

$$\sigma_l = \sqrt{H \cdot \tfrac{1}{2} \cdot \left(\tfrac{1}{\sqrt{H}}\right)^2} = \frac{1}{\sqrt{2}} \approx 0.707, \qquad (7)$$

which is a reasonable near-unit standard deviation that provides a good slightly conservative initialization of computed forward pass logits—crucially, without excessive input signal dilution—for fast stable convergence. The complete variance-balanced weight initialization strategy has been empirically confirmed at every layer of the model to work exactly like the statistical modeling suggests (also as a verification of the underlying independence assumptions), up to the expected approximations already discussed.

B. Object Noun Dictionary

The object noun dictionary was created based on a multitude of data sources, but most prominently the WordNet dictionary [47], GNU Collaborative International Dictionary of English [9], and the categories used by 35 different image classification and object detection/segmentation benchmarks. The benchmarks were divided into two groups, *object* and *general* benchmarks, depending on whether the class names were considered to be all valid object nouns or alternatively only used to provide frequency boosts, respectively. The steps used to generate the object noun dictionary were as follows:

1. All noun synsets in WordNet were collected that are part of one of the following 11 lexical classes (considered to be the only ones out of the 26 available that could *possibly* contain object nouns): animal, artifact, body, event, feeling, food, object, person, phenomenon, plant, or possession.

2. The entire WordNet synset hierachy was formatted as a tree and used to manually select nodes and sub-hierarchies considered to fit the definition of an object noun. 1598 manual synset specifications were recorded and used to select 23 942 synsets. This is the only step that required manual human intervention.

3. The chosen synsets were expanded to a working list of lemmas, and canonicalization was used to collect and deal with alternate spellings. A total of 42 981 canonical lemmas resulted.

4. 4105 unique object names were loaded from the *object* benchmarks and used to increase the number of canonical lemmas to 43 588.

5. A dataset of 1818 spelling equivalence mappings between American and British English were used to add additional spellings to the lemmas.

6. The GNU Collaborative International Dictionary of English was parsed and used to provide lemma noun alternates, and matches between singular and plural forms, especially for irregular nouns.

7. The WordNet plural noun exceptions list was used to deal with multiple singulars that share a plural form and multiple plurals that share a singular form.

8. A word inflection strategy was used to generate singular forms for nouns that had only appeared as plurals thus far, and the plural forms for all seen singulars.

9. The main spelling was resolved for each lemma with the use of language heuristics, and multiple lemmas that shared common spellings across singulars and plurals were merged. At this point there were 118 045 noun alternates in total across the now 42 919 lemmas.

10. The n-gram frequencies of the Google Web Trillion Word Corpus [8, 49] were used to assign frequencies to each individual alternate of each lemma.

11. 68 077 (non-unique) class names were loaded from the *object* and *general* benchmarks, and used to boost the frequencies of matching nouns in the dataset.

12. All noun frequencies were converted to an affine log-scale to better represent how often the corresponding nouns should be sampled during dataset generation.

The final object noun dataset has 65 210 singulars and 31 316 plurals for a total of 96 526 noun alternates that are associated with 42 919 unique canonicalized object nouns. The average log-scale frequency of each noun alternate is 1.5, resulting in an average log-scale frequency of 3.5 per object noun.

C. LLM-based Caption Generation

Given the complete object noun dictionary, an LLM was used to generate rich dynamic captions involving each of the object nouns in the dictionary. Based on the log-frequencies obtained from the object noun dictionary, ten times as many captions are generated independently for singular and plural forms of the noun, with at most 100 being generated for either category. OpenAI GPT-3.5 Turbo was used, specifically *gpt-3.5-turbo-1106* with a temperature of 0.8, maximum 1000 output tokens, and example prompts/responses as shown in Prompt 1. Ten captions are generated in every API call, and parsed using all available spelling variations of the object noun into appropriate caption templates. For instance, *"Two children playing with a bicycle in the yard"* is parsed and stored as *"Two children playing with a* $\{$*singular*$\}$ *in the yard"*. In cases of words where there are multiple spellings, like for example *orangutan*, this ensures that a caption template can be used in future to generate data for any desired

Prompt 1: Singular/plural caption generation examples

SYSTEM: You are a helpful AI assistant.
PROMPT: Generate a list of 10 image captions each containing the target object "bicycle" (singular). Use the style of the image captions for large language model pre-training. Don't use poetic language. Use a different sentence structure for each caption. Output the list in the following form: <nr>. <caption>
RESPONSE:
1. A red bicycle parked against a brick wall.
2. The bicycle resting under a tree in the park.
3. A woman riding a bicycle along the beach.
4. A vintage bicycle leaning against a lamppost.
5. Two children playing with a bicycle in the yard.
6. The bicycle locked to a metal railing outside the store.
7. A man carrying a folded bicycle up the stairs.
8. The bicycle lying on the grass with its wheels spinning.
9. A bicycle with a basket filled with colorful flowers.
10. The bicycle propped up against a wooden fence.

SYSTEM: You are a helpful AI assistant.
PROMPT: Generate a list of 10 image captions each containing the target object "bicycles" (plural). Use the style of the image captions for large language model pre-training. Don't use poetic language. Use a different sentence structure for each caption. Output the list in the following form: <nr>. <caption>
RESPONSE:
1. A row of bicycles parked by the street, waiting for their riders.
2. Bicycles lined up neatly in a bike-sharing station, ready for use.
3. Many bicycles secured to a rack outside a busy train station.
4. Several bicycles leaning against a wall, casting shadows in the afternoon sun.
5. Dozens of bicycles scattered across the open field during a cycling event.
6. A group of bicycles locked together in the urban bike parking area.
7. Bicycles parked in a row, their colorful frames catching the eye of passersby.
8. Multiple bicycles lined up near a city park, offering a convenient mode of transportation.
9. A cluster of bicycles parked haphazardly outside a bustling cafe.
10. Bicycles stacked on top of each other in a crowded bike storage area.

spelling if required by the sampling strategy. A total of 1.8M captions were generated in this way.

D. Effect of Noise Augmentation

Based on experiments, the 'magnitude' of Gaussian noise that performed best for noise augmentation during training is $G = 3.25$. The precise definition of how this noise was implemented was by adding elementwise Gaussian noise with a standard deviation of

$$\sigma_G = \frac{G}{\sqrt{F}} \tag{8}$$

to the input text embedding vectors and then renormalizing the result back to a unit vector. For the SigLIP B/16 model, which has an embedding dimension of $F = 768$, this corresponds to an elementwise standard deviation of $\sigma_G \approx 0.117$. If $\mathbf{g} = (g_1, \dots, g_F)$ is the random noise vector with $g_i \sim \mathcal{N}(0, \sigma_G^2)$, then we can see that

$$\mathbb{E}\big[\|\mathbf{g}\|^2\big] = \mathbb{E}\left[\sum_{i=1}^{F} g_i^2\right] = \sum_{i=1}^{F} \mathbb{E}[g_i^2] = F\sigma_G^2 = G^2, \tag{9}$$

$$\begin{aligned}\mathrm{Var}\big(\|\mathbf{g}\|^2\big) &= \mathrm{Var}\left(\sum_{i=1}^{F} g_i^2\right) = \sum_{i=1}^{F} \mathrm{Var}(g_i^2) = \sum_{i=1}^{F} 2\sigma_G^4 \\ &= 2F\left(\frac{G}{\sqrt{F}}\right)^4 = \frac{2G^4}{F}.\end{aligned} \tag{10}$$

For our case, this means the random noise vector has an average square-norm $\|\mathbf{g}\|^2$ of $G^2 \approx 10.56$ with a standard deviation of $G^2 \cdot \sqrt{\frac{2}{F}} \approx 0.539$. This clearly shows that statistically, the added noise vector will *always* have a norm significantly larger (on the order of $\|\mathbf{g}\| \approx G = 3.25$) than the unit embedding vector $\mathbf{e} = (e_1, \dots, e_F)$, which has $\|\mathbf{e}\| = 1$. On top of being such a large vector, due to the high dimensionality of the embedding space, the random noise vector $\mathbf{g}$ is actually also essentially *guaranteed* to be near-perpendicular to the embedding vector $\mathbf{e}$, causing a large directional change of the embedding vector no matter what—no embedding vector can ever remain anywhere close to where it started. To demonstrate this, we consider the cosine of the angle θ between $\mathbf{e}$ and $\mathbf{g}$, given by

$$\cos\theta = \frac{\mathbf{e}\cdot\mathbf{g}}{\|\mathbf{e}\|\|\mathbf{g}\|} = \sum_{i=1}^{F} e_i \frac{g_i}{\|\mathbf{g}\|}. \tag{11}$$

Due to the statistical independence of each element,

$$\begin{aligned}\mathbb{E}[\cos\theta] &= \mathbb{E}\left[\sum_{i=1}^{F} e_i \frac{g_i}{\|\mathbf{g}\|}\right] \\ &= \sum_{i=1}^{F} e_i \cancelto{0}{\mathbb{E}\left[\frac{g_i}{\|\mathbf{g}\|}\right]} \\ &= 0,\end{aligned} \tag{12}$$

due to the symmetrical isotropic Gaussian nature of $\mathbf{g}$. The corresponding variance is then

$$\begin{aligned}\mathrm{Var}(\cos\theta) &= \mathbb{E}[(\cos\theta)^2] - \cancelto{0}{\mathbb{E}[\cos\theta]^2} \\ &= \mathbb{E}\left[\left(\sum_{i=1}^{F} e_i \frac{g_i}{\|\mathbf{g}\|}\right)^2\right] \\ &= \sum_{i=1}^{F} e_i^2\, \mathbb{E}\left[\frac{g_i^2}{\|\mathbf{g}\|^2}\right] + \sum_{i\neq j}^{F} e_i e_j \cancelto{0}{\mathbb{E}\left[\frac{g_i g_j}{\|\mathbf{g}\|^2}\right]} \\ &= \sum_{i=1}^{F} e_i^2 \cdot \frac{1}{F} \\ &= \frac{1}{F},\end{aligned} \tag{13}$$

as $\mathbf{e}$ is a unit vector. In other words, the cosine similarity between an embedding vector and the noise vector that is added to it is on average 0 (perpendicular) with a very narrow standard deviation of $\frac{1}{\sqrt{F}} \approx 0.036$, implying essentially guaranteed near-perpendicularity. The average angle β between the initial embedding vector $\mathbf{e}$ and the noise augmented vector $\frac{\mathbf{e}+\mathbf{g}}{\|\mathbf{e}+\mathbf{g}\|}$ is thus approximately

$$\beta \approx \operatorname{atan}\left(\frac{\|\mathbf{g}\|}{\|\mathbf{e}\|}\right) \approx \tan^{-1} G \approx 72.9^\circ. \tag{14}$$

The standard deviation of the angle β can empirically be measured to be 1.94°. This narrow distribution of β, *i.e.* the distribution of angle separations that occur when using the nominal Gaussian noise augmentation strategy, is shown in Fig. 5. Also shown in the plot is the measured distribution of angle separations for *matching* and *non-matching* image-label pairs of the ImageNet-1K validation set, *i.e.* demonstrating the modality gap as well as the very thin separation that CLIP models provide between matching and non-matching pairs. The positive distribution-widening effect of the nominal strategy of adding 15% uniform angle noise from 45–75° is also shown (labeled "Gauss 3.25 + Uniform" in Fig. 5). With this nominal strategy, for 85% of all samples Gaussian noise of magnitude $G = 3.25$ is added, and for the remaining 15% of samples (randomly chosen), uniform angle noise of random magnitude 45–75° is applied.

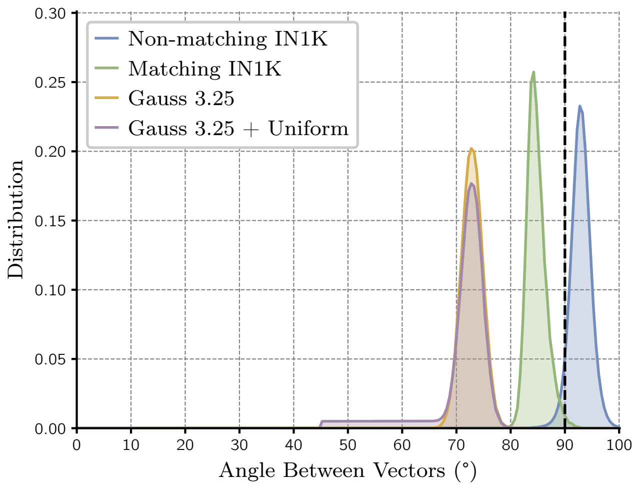


Figure 5. **Comparison of angle separation distributions.** A plot of the distribution of embedding vector angle separations in 768-dimensional embedding space for matching and non-matching image-label pairs from the ImageNet-1K validation set. Also shown is the distribution of angle separations between original and noise-augmented text embeddings for both the pure Gaussian and Gaussian with uniform strategies. The dashed vertical line shows exactly perpendicular vectors that are thus mathematically uncorrelated.

E. Open Vocabulary Image Datasets

Three open vocabulary image datasets were collected as part of this work, World, Wiki and Val3K, as described in the following sections. All three datasets were annotated both manually by humans and by a multimodal LLM (cf. App. F),[8] using one of five labels for each predicted object noun—*correct primary* (1.0 points), *correct secondary* (0.8 points), *close primary* (0.5 points), *close secondary* (0.4 points), and *incorrect* (0 points). The sum of all points is divided by the number of images to get the final percent accuracy *prediction score*. The suffixes -H and -L are used for the datasets to signify which annotations are being used, leading to five open vocabulary evaluation datasets, World-H, World-L, Wiki-H, Wiki-L, and Val3K-L.

E.1. World Dataset

The World dataset was collected with the express intention of trying to curate a dense collection of both regular and tricky object concepts. This included a specific focus on trying to include images from around the world that have never been on the internet (and thus cannot ever have been trained by a CLIP model before). Tricky images include ones that show: unusual objects, common object concepts but in peculiar variations, deceptive objects that are one thing but styled to look somewhat like another, and indirect representations of objects, *e.g.* paintings, cartoons, posters, sculptures, stuffed toys, drone show patterns, and such. A total of 12 people collected 85.3% of the images from original photographs in 10 countries of the world (Australia, Austria, England, France, Germany, Netherlands, New Zealand, Philippines, UAE, USA), with an active focus on covering as wide and varied concepts as possible. The remaining 14.7% of images were complemented from web sources to help cover concepts (*e.g.* animals) that could not so readily be manually photographed. The 272 images consist of 15.4% animals, 10.0% plants, 16.2% wider scenes, 6.6% indirect representations of objects, and 51.8% focused images of objects. Both human (World-H) and LLM (World-L) annotations are available for a wide variety of evaluated models.

[8] All except human annotations for Val3K.

E.2. Wiki Dataset

The Wiki dataset was collected by scraping 18 660 lead images, *i.e.* 'title images', from Wikipedia articles. Lead images often show a clear object concept that can be predicted and annotated, reducing the number of ambiguities and grey zones that need to be dealt with consistently during annotation. The scraped images had a noticeable bias in their topic of focus towards plants, animals, and people, so the sampling of the scraped images to a dataset size of 1 000 was guided by a probability-adjusted sampling method to conservatively correct that bias. Both human (Wiki-H) and LLM (Wiki-L) annotations were made for a variety of evaluated models.

E.3. Val3K Dataset

The Val3K dataset aims to help quantify how well a model is *really* doing on the ImageNet-1K validation set, if it is not forced to output lower probability classifications in order to align with the class names of the dataset. The dataset was collected by sampling 3 random validation images (out of the 50 available) for each of the 1 000 ImageNet-1K classes. This led to a dataset of 3 000 images of mixed content. LLM annotations were made on all 3 000 images to obtain Val3K-L. As LLM annotations, especially when passing images to a multimodal LLM, are attributed with cost, it was not feasible to perform annotations on the full 50 000 ImageNet-1K validation set.

F. Multimodal LLM Image Annotation

Due to the recent availability of some multimodal LLMs with a very high level of image understanding, it is conceivable to approximately quantify the performance of open vocabulary classification models by auto-annotating their predictions on image sets using these LLMs. This is particularly useful when it is just not feasible to perform the required amount of annotations as a human. The general process is as follows:

1. The chosen to-be-evaluated model(s) are inferenced on the required dataset of images, and the predicted object nouns are recorded (saving the top-k predictions for each image is also possible).
2. For each individual image, the deduplicated set of all predicted object nouns across all models is collected.
3. Groups of up to 15 of these predicted object nouns are evaluated per call to the LLM, until every object noun has up to 5 opinions from the LLM whether it is correct or incorrect (present in the image or not).
4. A ground truth annotation is established per-image per-predicted object noun based on the opinions that were given by the LLM.

We use OpenAI GPT-4o, specifically *gpt-4o-2024-05-13* with a temperature of 0.2, maximum 512 output tokens, and a cumulative probability threshold of 0.6 for nucleus sampling. The chosen parameters reflect the stability and consistency that is desired from the LLM. The prompt used for auto-annotation is shown in Prompt 2 for an example image. Asking the LLM to first provide a description of the image allows it to adjust its summarized knowledge of the image according to the provided object noun candidates, and enhances its classification precision. For best results, 8–15 object noun candidates are always given, as providing too few or too many has a small potential of causing inconsistent annotations for fine-grained object nouns. With intentional randomization of candidate ordering and which candidate is picked for which API request, each object noun is annotated 3–5 times, stopping before 5 if there is at least 80% agreement in the LLM's opinions. If the final agreement is at least 80%, then the corresponding opinion (*correct* or *incorrect*) is taken as the final annotation, otherwise the annotation is *close*. No *primary vs. secondary* classification is made because it was experimentally deemed too inconsistent. Due to the relatively unlikely condition required for the *close* category (LLMs tend to be wrong more often than they are inconsistent at a low temperature), it is systematically not seen as often as for human annotations.

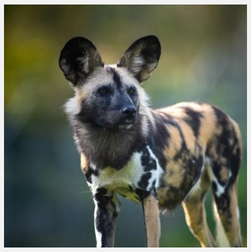
In order to verify that the LLM annotations are sufficiently accurate to make inferences about the performance of evaluated NOVIC models, we compared the prediction scores computed on human and LLM annotations for 397 models on the World dataset (cf. Fig. 6). As expected, a clear correlation and trend can be identified, whether *close* classifications are considered in the scores or not, with residual root mean square errors of 1.1% in both cases. This suggests that even though the absolute values of the scores do noticeably differ, the coarse ordering it applies to models is similar.

Prompt 2: Multimodal LLM image classification

SYSTEM: You are an AI assistant that has one and only one narrow task that you should strictly adhere to at all times. Given an image and an enumerated list of nouns by the user, you should first describe everything you see in the image in complete detail, and then provide an exactly matching enumerated list that for each noun provided by the user (explicitly repeat the noun) strictly classifies the noun into one of exactly two categories using a single word - Correct or Incorrect. Correct means that at least one instance of the noun is visible in the image. Incorrect means nothing very visually similar to the noun is visible in the image.

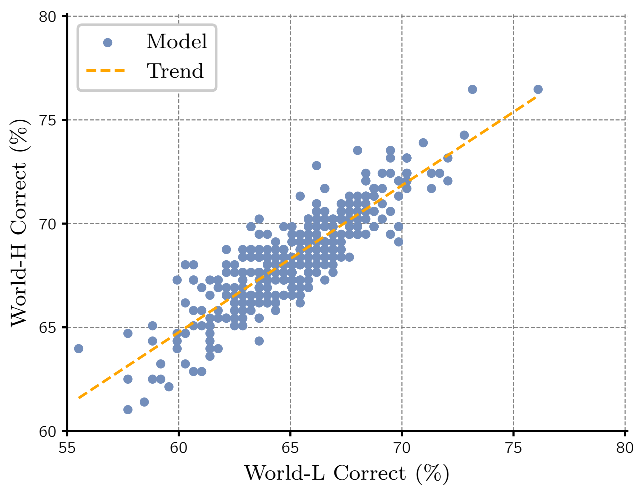
PROMPT: <*image base64*> (max 512×512)

1) african hunting dog
2) african wild dog
3) canis niger
4) cape hunting dog
5) hyena
6) hyena dog
7) lycaon pictus
8) mammal
9) mammalian
10) mexican hairless
11) mexican hairless dog
12) monkey dog
13) okapi



RESPONSE:
The image shows an African wild dog, also known as Lycaon pictus. It has a distinctive coat with patches of black, brown, white, and yellow, large rounded ears, and a slender body.

1) african hunting dog - Correct
2) african wild dog - Correct
3) canis niger - Incorrect
4) cape hunting dog - Correct
5) hyena - Incorrect
6) hyena dog - Incorrect
7) lycaon pictus - Correct
8) mammal - Correct
9) mammalian - Correct
10) mexican hairless - Incorrect
11) mexican hairless dog - Incorrect
12) monkey dog - Incorrect
13) okapi - Incorrect



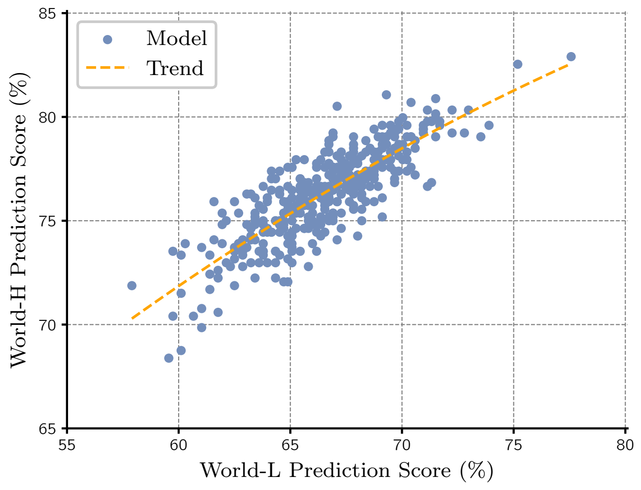


Figure 6. **Comparison of human and LLM image annotations.** World-H scores are plotted against World-L scores for 397 models. **Top:** Scores considering *correct* only. **Bottom:** Standard prediction scores considering *close* classifications as well. For human annotations, *secondary* classifications are scored as *primary* as the LLM does not make that distinction.

G. Model Hyperparameters

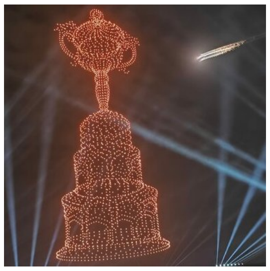
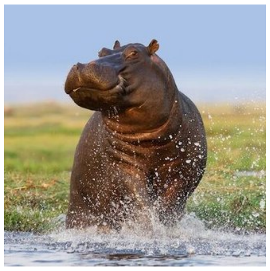
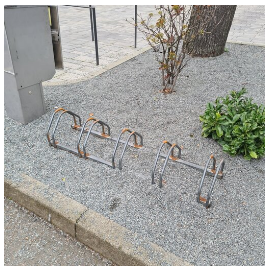
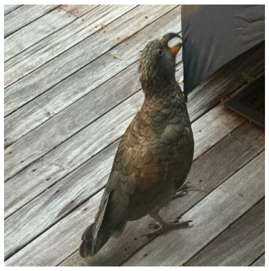
The nominal values of various NOVIC hyperparameters are shown in Tab. 5. The corresponding hyperparameter value ranges that were covered in some configurations during testing are also shown.

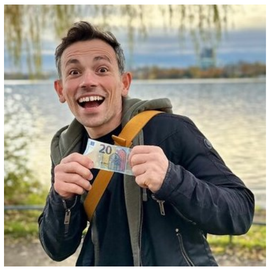
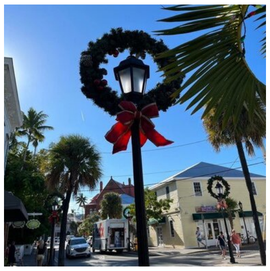
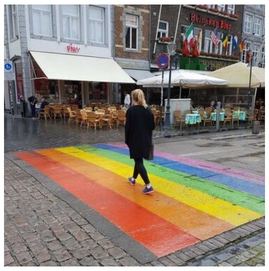
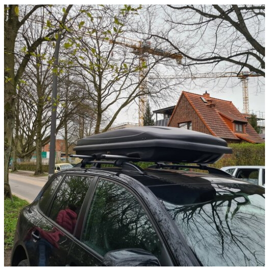
H. Qualitative Comparison

A qualitative comparison between NOVIC and image tagging baselines on the World-H and Wiki-H datasets is presented in Fig. 7 and Fig. 8, respectively. RAM and Tag2Text in general provide relatively coarse-grained classifications and often miss central concepts in the images, focusing instead on known nouns. For example, they output *man* for an image where a man is presenting a *banknote* to the camera.

Hyperparameter	Value	Tested Range
CLIP model	SigLIP B/16	See Tab. 4
CLIP mixed precision	float16	float16
Noun frequency threshold	0	0–10
Multiset order m	3	1–4
LLM captions	✓	✗/✓
Noise augmentation	Mix	Gauss, Uniform, Mix
Gauss noise magnitude	3.25	0.0–4.0
Uniform angle noise	45–75	Up to 0–85
Noise mix factor	15%	0–20%
Prefix tokens P	4	2–6
Hidden dim H	512	384, 512
Feedforward dim K	128	64–512
Num layers L	6	4–12
Num heads	8	8
Dropout	0.1	0.05–0.2
Bias parameters	✗	✗/✓
Label smoothing	✗	✗/✓
Beam width	10	1–10
Temperature	1	0.33–3
Epochs E	18	9–24
Batch size B	8192	512–16384
Optimizer	AdamW	AdamW, AdamP
Adam β_1	0.9	0.9
Adam β_2	0.95	0.95, 0.99
LR scheduler	cosine	cosine
LR warmup epochs	0	0–2
Initial LR	1.5e-3	5e-4 – 3e-3
Weight decay	0.1	0–0.3
Weight decay for 1D	✗	✗/✓

Table 5. **NOVIC hyperparameter values and tested ranges.** Batch sizes use gradient accumulation up to factor 16.

				
Tag2Text	stage (I)	water (S)	ladder (I)	bird (P)
RAM	stage (I)	water (S)	pavement (S)	bird (P)
RAM (FT0)	stage (I)	water (S)	pavement (S)	bird (P)
SigLIP NOVIC (FT2)	trophy cup (P)	hippopotamus amphibius (P)	bicycle rack (P)	parrot (CP)
SigLIP NOVIC (FT0)	trophy cup (P)	hippopotamus amphibius (P)	bicycle rack (P)	mourning dove (I)
DFN-5B NOVIC (FT0)	firework (S)	hippopotamus (P)	bicycle rack (P)	kea (P)

				
Tag2Text	man (P)	palm tree (S)	walk (I)	car (P)
RAM	man (P)	palm tree (S)	walk (I)	car (P)
RAM (FT0)	man (P)	palm tree (S)	female person (P)	car (P)
SigLIP NOVIC (FT2)	money (P)	wreath (P)	pavement (P)	roof rack (P)
SigLIP NOVIC (FT0)	money (P)	wreath (P)	pedestrian crossing (P)	roof rack (P)
DFN-5B NOVIC (FT0)	banknote (P)	florida key (P)	pedestrian (P)	roof rack (P)

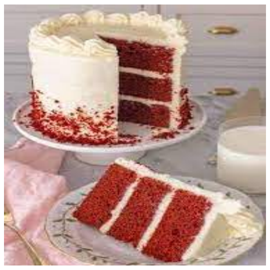
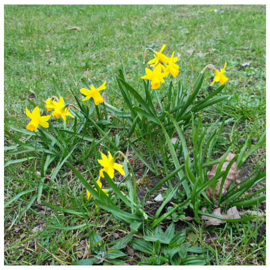
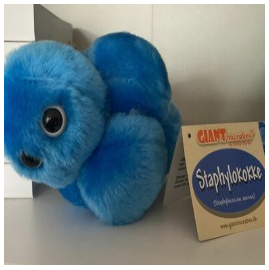
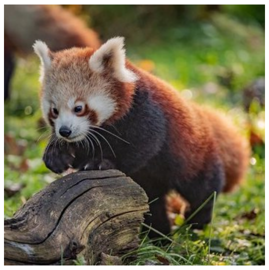
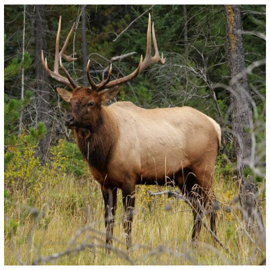
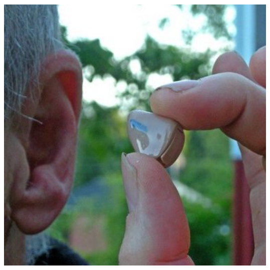
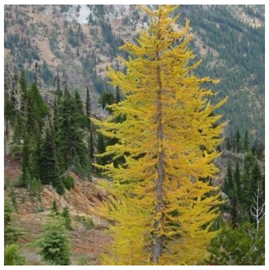
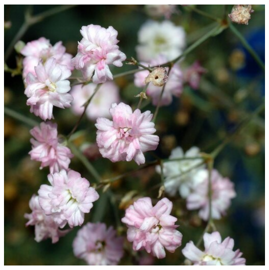
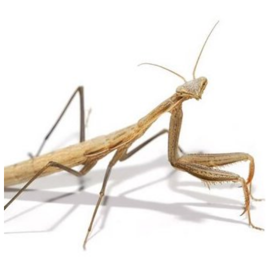
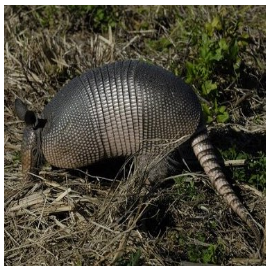
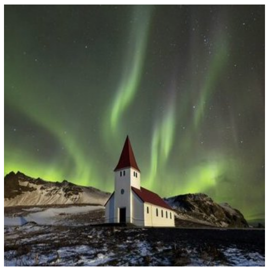
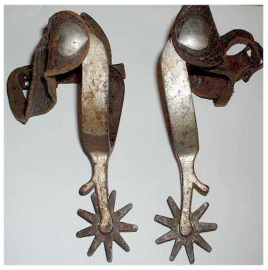
				
Tag2Text	cake (P)	flower (P)	animal (I)	panda (P)
RAM	cake (P)	yellow (I)	blue (I)	red (I)
RAM (FT0)	cake (P)	flower (P)	blue (I)	red cole (I)
SigLIP NOVIC (FT2)	red velvet cake (P)	jonquil (P)	stuffed toy (P)	red panda (P)
SigLIP NOVIC (FT0)	cake cake (I)	jonquil (P)	stuffed toy (P)	red panda (P)
DFN-5B NOVIC (FT0)	red velvet cake (P)	daffodil (P)	genus staphylococcus (P)	ailurus (P)

Figure 7. **Examples of open vocabulary classification on the World-H dataset.** While RAM and Tag2Text select the most likely noun from a limited vocabulary, NOVIC can freely output any object noun. Possible classification scores are *correct primary* (P), *correct secondary* (S), *close primary* (CP), *close secondary* (CS), and *incorrect* (I). Correct primary and incorrect labels are highlighted in color. Output labels that are not nouns are uninformative for the image and thus considered incorrect. Similarly, invalid nouns are also deemed incorrect.

				
Tag2Text	elk (P)	man (S)	tree (P)	flower (P)
RAM	moose (CP)	hand (S)	tree (P)	flower (P)
RAM (FT0)	stand (I)	finger (S)	tree (P)	flower (P)
SigLIP NOVIC (FT2)	buckskin (I)	hearing aid (P)	brewers spruce (CP)	babies breath (P)
SigLIP NOVIC (FT0)	moose wood (I)	hearing aid (P)	western larch (CP)	genus saponaria (CP)
DFN-5B NOVIC (FT0)	elk (P)	hearing aid (P)	subalpine larch (P)	genus gysophila (P)

				
Tag2Text	mantid (P)	grass (CS)	church (P)	old (I)
RAM	photo (I)	armadillo (P)	church (P)	tool (P)
RAM (FT0)	mantis (P)	armadillo (P)	church (P)	tool (P)
SigLIP NOVIC (FT2)	praying mantis (P)	nine banded armadillo (P)	church (P)	spur (P)
SigLIP NOVIC (FT0)	praying mantis (P)	armadillo (P)	church (P)	spur (P)
DFN-5B NOVIC (FT0)	mantis (P)	dasypus (P)	aurora (P)	spur (P)

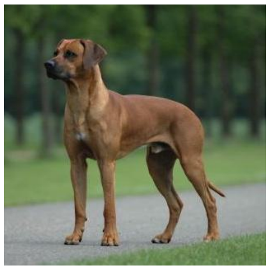
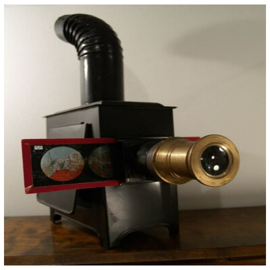
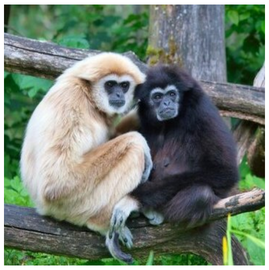
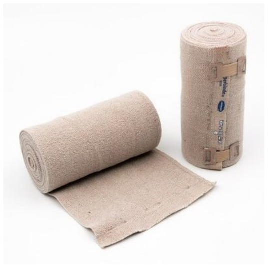
				
Tag2Text	dog (P)	telescope (CP)	monkey (P)	roll (P)
RAM	dog (P)	camera (CP)	sit (I)	roll on (I)
RAM (FT0)	dog (P)	camera (CP)	monkey (P)	roll (P)
SigLIP NOVIC (FT2)	rhodesian ridgeback (P)	viewfinder (I)	gibbon (P)	plaster bandage (P)
SigLIP NOVIC (FT0)	rhodesian ridgeback (P)	viewfinder (I)	hylobates syndactylus (CP)	compression bandage (P)
DFN-5B NOVIC (FT0)	rhodesian ridgeback (P)	magic lantern (P)	gibbon (P)	compression bandage (P)

Figure 8. **Examples for open vocabulary classification on the Wiki-H dataset.** The images are an assortment of Wikipedia lead images, covering a broad range of visual concepts. Possible classification scores are *correct primary* (P), *correct secondary* (S), *close primary* (CP), *close secondary* (CS), and *incorrect* (I). Correct primary and incorrect labels are highlighted in color.

Configuration	World-H	Wiki-L	IN1K
Prefix tokens $P = 2$	77.91	62.75	65.51
Prefix tokens $P = 4$*	**78.92**	63.97	**68.11**
Prefix tokens $P = 6$	78.03	**64.15**	68.08
Layers $L = 4$	77.23	62.38	63.85
Layers $L = 6$*	**78.92**	63.97	68.11
Layers $L = 8$	77.94	62.78	69.05
Layers $L = 12$	78.15	**64.00**	**71.19**
Hidden dim $H = 384$	76.97	61.65	65.77
Hidden dim $H = 512$*	**78.92**	**63.97**	**68.11**
Batch size $B = 4096$	78.14	63.73	66.76
Batch size $B = 8192$*	78.92	**63.97**	**68.11**
Batch size $B = 16384$	**79.56**	63.17	67.85
Epochs $E = 12$	77.33	63.30	66.33
Epochs $E = 18$*	**78.92**	**63.97**	**68.11**
Epochs $E = 24$	77.77	63.00	67.71

Table 6. **Object decoder architecture and training hyperparameter ablations (% mean of 3).** IN1K is the ImageNet-1K classification performance of the trained decoders, and the asterisks correspond to the nominal ablation baseline (FT2). World-H and Wiki-L are prediction scores (described in Sec. 4) on the corresponding datasets. See Sec. 4.2 for more ablation results.

Additionally, they sometimes output colors (*e.g. blue*, *yellow*), verbs (*e.g. walk*, *stand*), attributes (*e.g. old*), or merely image style information (*e.g. photo*), thereby failing to convey the actual content of the image. Even when given access to the complete object noun dictionary (*i.e.* FT0), which is an order of magnitude more comprehensive than the RAM vocabulary, RAM predominantly predicts the familiar words it was trained on rather than utilizing a broad range of object nouns as labels.

On the other hand, NOVIC demonstrates strong capabilities in precisely detecting and outputting the central object of an image. Particularly when trained on the DFN-5B CLIP model, NOVIC provides very fine-grained classifications and can accurately identify Latin names of plants and animals (*e.g. genus gypsophila*), names of very specific uncommon objects (*e.g. magic lantern*), and even geolocations of pictures (*e.g. florida key*) with high precision. Since NOVIC generates free-form text during inference, it occasionally fails to "stop in time" when outputting nouns however, resulting in incorrect labels (*e.g. cake cake*, *moose wood*). This issue is more prevalent for smaller CLIP models and lower word frequency thresholds of the training dataset.

I. Decoder Architecture Ablations

Ablations of the core NOVIC model hyperparameters are shown in Tab. 6. The results support the nominal choice of hyperparameters, and suggest that slight gains could potentially be made with a greater number of transformer layers (although model generalization becomes slightly less reliable) and/or batch size. Refer to Sec. 4.2 for ablation results pertaining to the training dataset, noise augmentation, and CLIP model.

J. Image Classification Benchmark Trends

Tab. 1 shows the performance of NOVIC on various image classification benchmarks, many of which are commonly used to benchmark the zero-shot image classification performance of CLIP models. In general, the trend can be seen that the classification performance of NOVIC models slowly drops off as the complexity and extensiveness of the training data is increased towards the full size of the object noun dictionary. This can be expected, as the object decoder has increasingly many object concepts to learn as the frequency threshold (FT) decreases, and these concepts are also increasingly densely packed in the embedding space, making them harder to accurately keep apart. An exception to this general trend for some of the classification benchmarks is the FT9 score, as the FT9 training data does not actually always cover all of the target nouns required for each dataset. This concretely means that $412/1000$ ImageNet-1K classes, $17/200$ Tiny ImageNet classes, $50/200$ ImageNet-A classes, and $37/200$ ImageNet-R classes, cannot contribute to the final classification score, resulting in the disadvantaged lower scores marked with a dagger (†) in Tab. 1. Another exception to this general trend are the benchmarks that have few and coarse-grained classes, such as Imagenette and CIFAR-10. For these datasets, the performance depends less on the diversity of object nouns seen during training, and more on just the raw ability of the CLIP image encoder to produce an accurate and representative embedding, resulting in fairly consistent classification performances irrespective of the frequency threshold.

Training on large-scale datasets is also inherently not deterministic—in particular also due to the randomized data (*e.g.* noise) augmentation—and some natural variations in the results will occur even if taking the mean of three training runs. The object decoder has thousands to tens of thousands of concepts to learn, and evaluating on a classification dataset with *e.g.* 10–101 classes is probing a small fraction of what the model has learned and is rewarded for by its training loss. This means that two models with identical loss will always have some concepts that they have learned better, and some they have not learned quite as robustly. If that happens to align with the classes required for a classification dataset, the models then either do slightly better or slightly worse on them. As previously mentioned, CIFAR-10 is an example of all models doing about the same, up to natural variation, as the dataset classes are very few and coarse, and the dataset is relatively 'easy'.

The classification results for the FT6 and FT2 NOVIC

Model	Prompt-free Classification	Text-only Training	Seen Objects	Used Objects	RAM Vocab	Primary Score		Prediction Score		Overall Score
Tag2Text [29]	✗	✗	3 429	176	91.9	62.32	$\xleftarrow{-15.29}$	77.61	$\xrightarrow{-13.49}$	64.12
RAM [76]	✗	✗	4 585	160	100.0	57.35	$\xleftarrow{-17.65}$	75.00	$\xrightarrow{-15.86}$	59.14
RAM (FT2)	✗	✗	11 897	188	76.8	64.71	$\xleftarrow{-11.61}$	76.32	$\xrightarrow{-13.84}$	62.48
RAM (FT0)	✗	✗	**42 919**	188	74.3	63.97	$\xleftarrow{-11.03}$	75.00	$\xrightarrow{-13.84}$	61.16
NOVIC (FT9)	✓	✓	2 919	235	66.9	74.82	$\xleftarrow{-2.79}$	77.61	$\xrightarrow{-3.22}$	74.39
NOVIC (FT6)	✓	✓	5 899	252	58.5	76.47	$\xleftarrow{-2.21}$	78.68	$\xrightarrow{-1.01}$	77.67
NOVIC (FT2)	✓	✓	11 897	253	52.6	77.94	$\xleftarrow{-1.62}$	79.56	$\xrightarrow{-0.64}$	78.92
NOVIC (FT0)	✓	✓	**42 919**	251	49.6	71.14	$\xleftarrow{-3.53}$	74.67	$\xrightarrow{-1.10}$	73.57
NOVIC† (FT2)	✓	✓	11 897	**263**	55.5	86.77	$\xleftarrow{-1.17}$	87.94	$\xrightarrow{-0.81}$	87.13
NOVIC† (FT0)	✓	✓	**42 919**	260	**48.5**	**86.95**	$\xleftarrow{-1.32}$	**88.27**	$\xrightarrow{-0.37}$	**87.90**

Table 7. **Open vocabulary World-H results in comparison to baselines (best of 3).** Tag2Text and RAM achieve artificially high scores by frequently outputting uninformative generic tags like *man*, *tree*, *grass*, *plate*, and often miss the actual intent of an image. Scores are in %. *RAM Vocab* is the percent of samples whose predictions are in the core RAM vocabulary. *Primary Score* is the *Prediction Score* when rewarding only primary predictions. *Overall Score* is the *Prediction Score* with $\times 0.5$ applied to the score contributions of any coarse predictions, effectively penalizing vague generic predictions. See Tab. 3 for Wiki-H results. †Using DFN-5B H/14-378 CLIP model.

models are similar for many of the benchmarks, as both training sets have a reasonable size and comfortably cover all required class names as target nouns. The dip in performance for FT0 is more pronounced in general, due to the near-quadrupling relative to FT2 of the number of object concepts to learn, with the magnitude of the dip being tendentially correlated to larger numbers of classes and more fine-grained classification concepts overall. Both of these factors result in smaller regions of the embedding space, with less cosine distance between them, needing to be kept apart. This, in combination with the more finely 'concept-subdivided' embedding space of an FT0 model, leads to the performance dip.

K. Additional Baseline Comparisons

In Tabs. 3 and 7, we compare (as described in Sec. 4.1) NOVIC to the image tagging baselines Tag2Text [29] and RAM [76], on the Wiki-H and World-H datasets respectively. While the image tagging baselines require a list of noun candidates to be supplied at inference time and are trained on image-text pairs, NOVIC performs completely prompt-free image classification and is trained much more efficiently on text-only data. Also, while NOVIC is trained primarily on a synthetic text dataset constructed in a controllable and unbiased manner from an entire English dictionary of object nouns, the baselines train on 'only' 14M image-text pairs, which have a significantly lower diversity of seen objects, and do not offer any controllability of the distributions of the nouns and image embeddings seen during training. Furthermore, Tag2Text has a fixed set of labels, while RAM attempts to provide some level of open vocabulary recognition beyond its nominal set of labels by distilling an image encoder from the base CLIP image encoder. Due to the limited size of the used image-text dataset however, and noting that the core set of labels was already *constructed* by parsing these exact text captions, one can understand that the 'open' abilities of RAM are somewhat limited, *i.e.* RAM is not truly open vocabulary in any way close to how one considers CLIP to exhibit open vocabulary abilities.

In Tabs. 3 and 7, we can see that NOVIC has seen significantly more diverse object nouns during training (42 919 as opposed to less than a few thousand for the baselines), and this is clearly reflected by its ability to use a wide array of object nouns in its predictions—generally close to as many unique nouns as there are images in the corresponding dataset (*e.g.* 263 unique predictions for 272 images). The tendency of the baselines not to provide such a diverse set of predictions can be observed from their significantly lower number of used objects. The restricted ability of RAM to perform open vocabulary classifications can also be identified by observing the high proportion of its generated predictions that belong to the core RAM vocabulary, even when provided with a large variety of object nouns to choose from (*e.g.* FT0). RAM's open vocabulary predictions are also observed to have a significantly higher error rate than core vocabulary predictions.

Three prediction scores are calculated for each model—*Prediction Score*, the standard prediction score as defined in Sec. 4, *Primary Score*, which does not reward secondary predictions and thereby emphasizes the important ability of the model to focus on the actual intended object of an image,[9] and *Overall Score*, which adjusts the standard prediction

[9]Note that even the primary score does not adequately penalize the baselines' scores in practice however, for example in instances where a plate of food is the focus of the image and the baselines invariably predict *plate* whereas NOVIC tends to predict the specific name of the dish.

score to reduce the reward for overly coarse predictions and thereby emphasizes the important ability of the model to provide meaningful and informative fine-grained classifications. It should be noted that both the primary and overall scores can by definition only at most be equal to the standard prediction score, so an ideal model that emits only *fine-grained primary* predictions would have all three scores be equal. We can observe that for NOVIC this is indeed almost the case, whereas the baselines exhibit very significant drops in score when penalizing secondary and coarse predictions as such. Overall, this demonstrates that NOVIC produces high quality predictions that are simultaneously diverse and fine-grained yet at the same time accurate, as well as correctly focusing on the main object of an image, all while being real-time, prompt-free, truly open vocabulary, and very efficiently trained with text only. This makes NOVIC a strong improvement over the state-of-the-art baselines.

L. Ethical AI

Our open vocabulary image classifier may inherit biases from the CLIP model it is trained on, and be similarly susceptible to adversarial attacks during inference, such as manipulated images misguiding label generation. Although the object decoder is generative, it is trained in a constrained setting, making it unlikely to produce sensitive outputs. Training the object decoder on the proposed text datasets is significantly more cost-effective than training alternative methods such as RAM on large-scale image datasets. However, due to the large amount of noise used, identifying the optimal model configuration often necessitates many training runs, each with an environmental impact.