Learning from Semantic Dictionaries: Discriminative Codebook Contrastive Learning for Unified Visual Representation and Generation

Supplementary Material

The supplementary material provides additional setup and hyperparameter details, extended transfer learning results, deeper analyses of the generative and discriminative codebooks, dense-task evaluations, and qualitative generation results.

A. Experiment setup

For both generative and discriminative tasks, we follow the experimental setup as in MaskGIT [7], MAGE [32] and Sorcen [19]. In tables A.1 to A.5, we show the most relevant hyperparameters for all experiments. We provide the robustness dataset details in Table A.6. Note that, for conditional generation setup (Table A.5), we do *not* retrain the decoder, but fine-tune it exclusively on masked token reconstruction tasks. For guidance, we concatenate the extracted discriminative centroids of every patch and the CLIP class embedding to the main reconstruction canvas that is later fed to the decoder. The encoder and rest of the elements in the architecture is kept frozen during this process. For all pretraining, the training dataset is precomputed before the training. This process can be done in $\sim$7 hours on a single H100, including the k-Means computation required for the discriminative codebook creation. Note that this computation is done only once and can be considered negligible when compared against the $\sim$951 hours introduced by the online VQGAN tokenizer used in MAGE [22].

B. Hyperparameter Ablations

We ablate the two main hyperparameters of the codebook contrast objective: the weighting factor λ and the number of neighbor centroids. These ablations are performed on IN200 subsample dataset and trained for 200 epochs. As shown in Table B.7a $\lambda = 0.1$ yields the best results, concurring with previous works [19, 26, 32]. For the number of neighbor centroids, we observe that using 5 or 30 neighbors yields the strongest generative performance. However, the larger set of 30 neighbors reduces the model's discriminative capacity. Overall, LEASE remains robust across a broad range of neighbor counts, with 5 neighbors offering the best balance, consistent with findings from prior neighbor-based contrastive frameworks [18, 27].

C. Extended Transfer Learning Results

In Table C.8 and Table C.9, we extend transfer learning experimentation to 8-shot and 4-shot regimes. In the 8-shot scenario, LEASE demonstrates the strongest overall performance, achieving the highest average accuracy across datasets and outperforming previous VQGAN-based methods [19, 32] in 6 out of 7 datasets. In the 4-shot setting LEASE maintains a clear advantage, showing the highest average accuracy among the three methods and outperforms both competitors on all datasets. As shown in 16-shot setting (Refer Table 4 from the main manuscript), LEASE consistently shows superior few-shot transfer performance, while 8-shot and 4-shot settings prove that this margin increases as the number of shots decreases.

config	value
optimizer	AdamW
base learning rate	1.5e-4
weight decay	0.05
optimizer momentum	β_1, β_2 = 0.9, 0.95
batch size	4096
learning rate schedule	cosine decay
warmup epochs	40
training epochs	1600
gradient clip	3.0
label smoothing	0.1
dropout	0.5
masking ratio min	0.5
masking ratio max	1.0
masking ratio mode	0.55
masking ratio std	0.25
λ	0.1
NN number	5
τ	0.1
α	0.1

Table A.1. Pre-training Settings.

config	value
optimizer	LARS
base learning rate	0.1
weight decay	0
optimizer momentum	0.9
batch size	4096
learning rate schedule	cosine decay
warmup epochs	0
training epochs	90
augmentation	RandomResizedCrop

Table A.2. Linear Probing Settings.

config	value
optimizer	AdamW
base learning rate	2.5e-4
weight decay	0.05
optimizer momentum	$\beta_1, \beta_2 = 0.9, 0.999$
layer-wise lr decay	0.65
batch size	1024
learning rate schedule	cosine decay
warmup epochs	5
training epochs	100
label smoothing	0.1
augmentation	RandAug (9, 0.5)
mixup	0.8
cutmix	1.0
random erase	0
drop path	0.1

Table A.3. End-to-End Finetuning Settings.

config	value
optimizer	LARS
base learning rate	1.0
weight decay	0.0
optimizer momentum	0.9
batch size	16
learning rate schedule	cosine decay
warmup epochs	0
training epochs	10
augmentation	RandomResizedCrop

Table A.4. Few-shot Settings.

config	value
optimizer	AdamW
base learning rate	1.5e-4
weight decay	0.05
optimizer momentum	β_1, β_2 = 0.9, 0.95
batch size	4096
learning rate schedule	cosine decay
warmup epochs	5
training epochs	300
gradient clip	3.0
label smoothing	0.1
dropout	0.5
masking ratio min	0.5
masking ratio max	1.0
masking ratio mode	0.55
masking ratio std	0.25

Table A.5. Conditional Finetuning Settings. Note that only the decoder is finetuned.

dataset	corruption / specialty
ImageNet-1k (val)	clean validation
ImageNet-V2 (INv2)	matched-frequency test
ImageNet-Sketch (IN-S)	sketch-based domain shift
ImageNet-Rend. (IN-R)	artistic renditions
ImageNet-A (IN-A)	adversarial examples
ObjectNet (ObjN.)	object distribution shift
ImageNet-C	multi-corruption

Table A.6. Datasets and their corresponding corruption types or domain specializations.

λ value	LP	FID	IS
0.1	85.47	16.33	58.11
0.5	85.43	17.31	56.31
1.0	84.93	17.56	56.21

(a) Contrastive objective weight.

# NN	LP	FID	IS
5	85.47	16.33	58.11
15	85.50	16.80	57.25
30	85.36	16.31	58.29

(b) Number of neighbors ablation.

Table B.7. Ablation results for linear probing accuracy (LP), FID and IS.

	Caltech	UCF101	Sun	DTD	C100	C10	Places	Avg.
MAGE	83.94	46.37	43.38	40.13	53.08	**75.02**	27.00	52.70
Sorcen	85.31	**50.70**	44.65	39.60	49.73	65.02	27.32	51.76
LEASE	**87.06**	**50.91**	**47.86**	**42.02**	**56.60**	72.99	**31.12**	**55.51**

Table C.8. Transfer learning results (Top-1 accuracy (%)) for different datasets under 8-shot settings.

	Caltech	UCF101	Sun	DTD	C100	C10	Places	Avg.
MAGE	71.72	29.71	31.36	19.92	40.03	35.06	19.81	35.37
Sorcen	69.20	36.35	33.48	21.99	35.58	34.44	20.32	35.91
LEASE	**78.22**	**39.62**	**36.77**	**27.01**	**47.42**	**44.04**	**24.72**	**42.54**

Table C.9. Transfer learning results (Top-1 accuracy (%)) for different datasets under 4-shot settings.

D. Generative vs. Discriminative Codebook

Relationship between Codebook Pairs. We analyse how generative (GEN) and discriminative (DISC) codebooks relate by computing the conditional entropies $H(GEN|DISC)$ and $H(DISC|GEN)$ from the patch-aligned token co-occurrence matrix. This conditional entropy $H(X|Y)$ measures how uncertain the value of a random variable X remains when another variable Y is known. In practice, it quantifies the average amount of information required to describe X after knowing Y. Low conditional entropy means knowing Y makes X highly predictable. Figure D.1 shows the distribution of these entropies across all tokens (measured in nats), and reveals a strong asymmetry between the two directions.

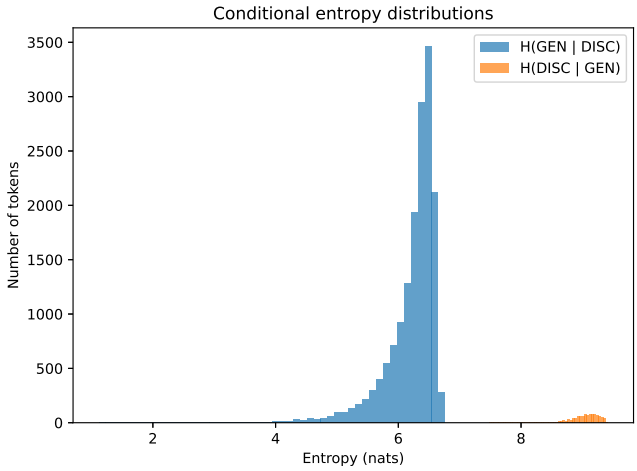

Figure D.1. Conditional entropy distributions for discriminative and generative codebooks.

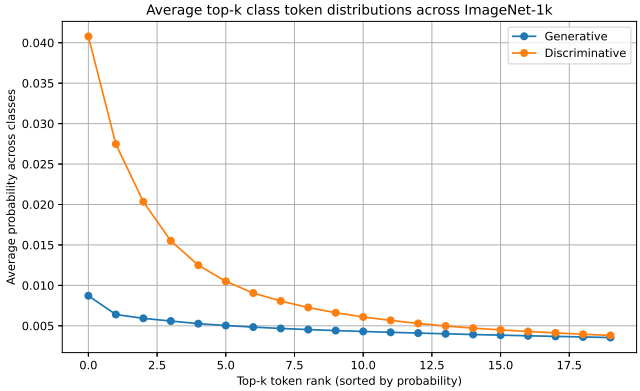

Figure D.2. Average top-k token distributions across all ImageNet classes.

A subset of discriminative tokens leads to low $H(GEN|DISC)$, meaning that these discriminative tokens consistently co-occur with a small set of generative tokens. In contrast, $H(DISC|GEN)$ is uniformly high, indicating that generative tokens provide almost no information about the corresponding discriminative tokens. This reflects the fundamental difference between the two codebooks: the generative codebook, based on VQGAN, is texture-based and largely class-agnostic, whereas the discriminative codebook, made by the features of DINOv2, encodes semantic distinctions learned through local appearance signals. Consequently, even if discriminative tokens do not explicitly encode textures, their semantic prototypes are tied to characteristic visual patterns that allow generative tokens to be predicted in some cases. This observation is consistent with prior work such as DiGIT [61], which is able to train a generative model solely from a codebook based on DINOv2. Still, the low-entropy behaviour only occurs for a small amount of discriminative tokens ($\sim$500 out of 16K), while the majority produce entropies above 6 nats. This indicates that the semantic overlap between the two codebooks is limited: most discriminative tokens correspond to a broad set of VQGAN textures, and generative tokens do not encode the semantic distinctions present in the discriminative codebook.

Class-average Token Distributions. We display the semantical contrast between the generative and discriminative codebooks used to train LEASE by computing the top-k token probabilities for each ImageNet class and averaging them across all 1000 classes. In practice, we extract the frequency of every encoded patch token on all ImageNet classes for both codebooks. Most frequent tokens for a given class are assumed to contain more discriminative information about that specific class. At the same time, the higher the frequency the higher the discriminative information, as it would mean that an specific class is better represented by the token. These frequencies are represented as a probabilities for every class on ImageNet and averaged. In Figure D.2, we can see the average probability of the top 20 most probable tokens per class.

The discriminative codebook exhibits a sharp, heavy-tailed distribution, where a small number of tokens dominate the class representation. This indicates strong class-specific semantics. In contrast, the generative codebook produces an almost flat curve, with top tokens only marginally more probable than lower ones. This marginality, combined with the masked token reconstruction strategies, could explain the discriminative capacity of works such as MAGE [32] and Sorcen [19], which leverage a generative codebook exclusively. Still, this behavior confirms that generative tokens carry little class-specific information and mainly encode generic textures. The aggregation over all classes demonstrates that this behavior is global rather than class-specific. The discriminative codebook, even if it comes from a self-supervised model, consistently organizes patches into semantically meaningful groups, while the generative codebook remains largely class-agnostic.

Token-level Shared Structure. Finally, to visualize the overlap between the two codebooks, we compute the pointwise mutual information (PMI) between every generative and discriminative token and display a heatmap for the pairs with highest PMIs (40 tokens per axis in total) in Figure D.3. Each column corresponds to one discriminative token and each row to one generative token. We observe only a few isolated bright cells per column, indicating that a small number of discriminative tokens consistently co-occur with some specific generative tokens. The majority of entries are close to zero PMI, even in this top-PMI subset,

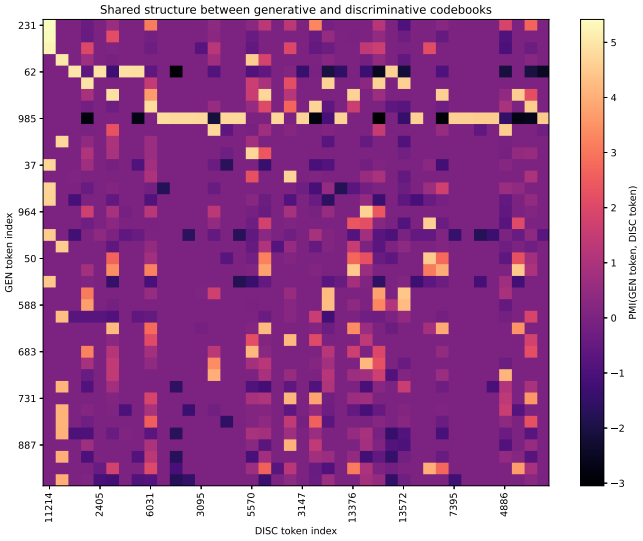

Figure D.3. PMI heatmap between the generative and discriminative codebooks.

and we do not see large block structures that would indicate a shared latent organization. This confirms that the overlap between the generative and discriminative codebooks is sparse. Individually, both codebooks contain useful information for their respective domains, either generation or discrimination. However, these semantics are not shared among both. The limited overlap is not simply an effect of vocabulary size, but rather reflects a fundamental difference in what the two codebooks represent: the discriminative codebook organizes patches according to semantic object structure, while the generative codebook captures texture-based, class-agnostic appearance.

E. Dense Task Evaluation

Token-based approaches, including MAGE [32] and Sorcen [19], continue to face limitations on dense prediction tasks due to the loss of fine-grained spatial information [47] caused by the quantized input tokens. As shown in Table E.10, all methods achieve comparable performance on MSCOCO, with Sorcen and LEASE slightly improving over MAGE. On FoodSeg, a more specific dataset, Sorcen stands as the best while LEASE matches the performance obtained by MAGE.

	MSCOCO	FoodSeg
MAGE	15.80	15.88
Sorcen	15.90	16.82
LEASE	15.92	15.85

Table E.10. Extended evaluation on Instance Segmentation downstream task. mAP metric is reported.

F. Generation Visualization

To further illustrate the generative capabilities of LEASE, we present qualitative samples under both unconditional and class-conditional settings in Figure F.4 and Figure F.5. Additionally, we also visualize the reconstruction, inpainting and outpainting capacity of LEASE in Figure F.6 and Figure F.7.

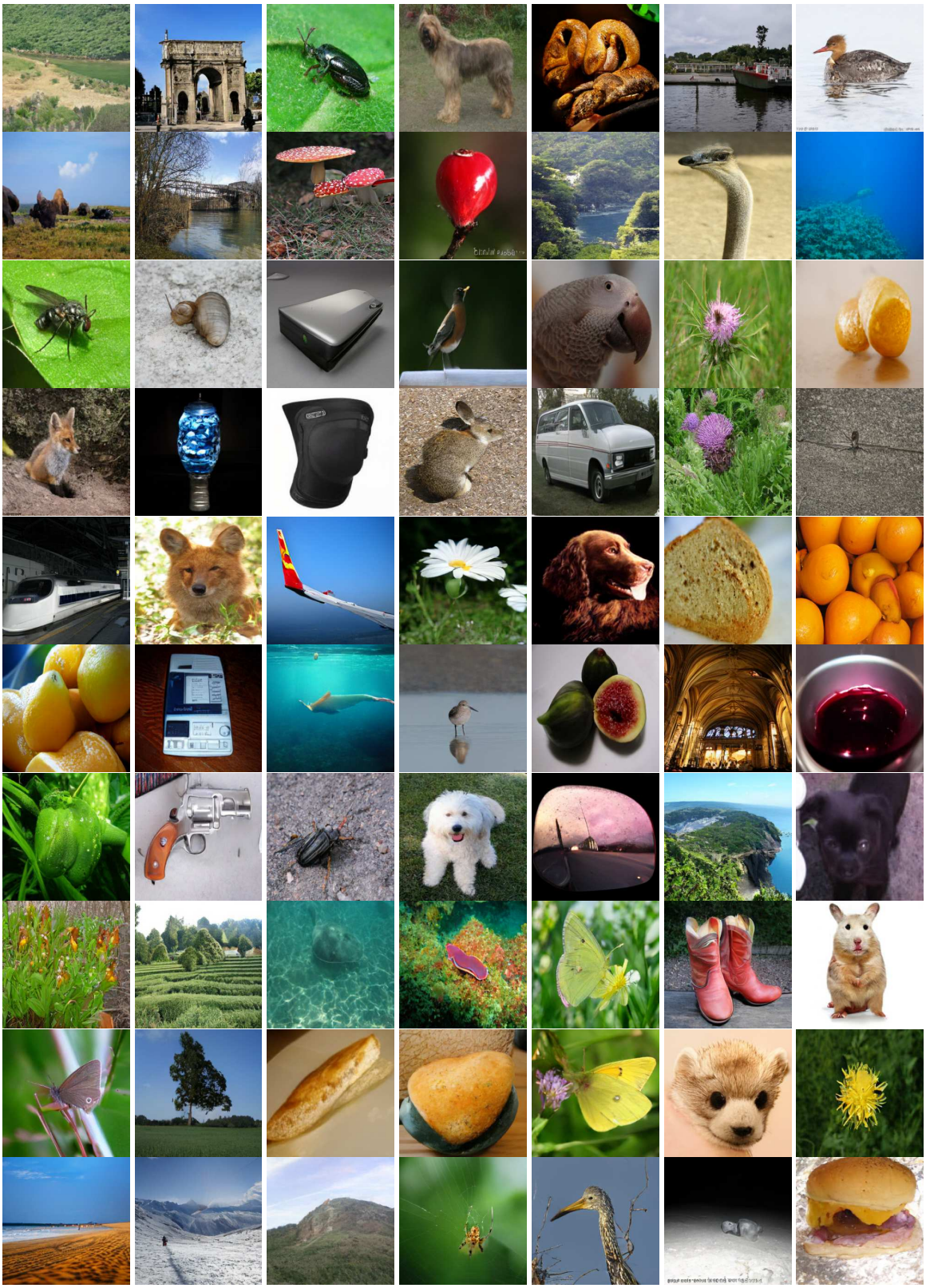

Figure F.4. Unconditional generation examples of LEASE using ViT-B.

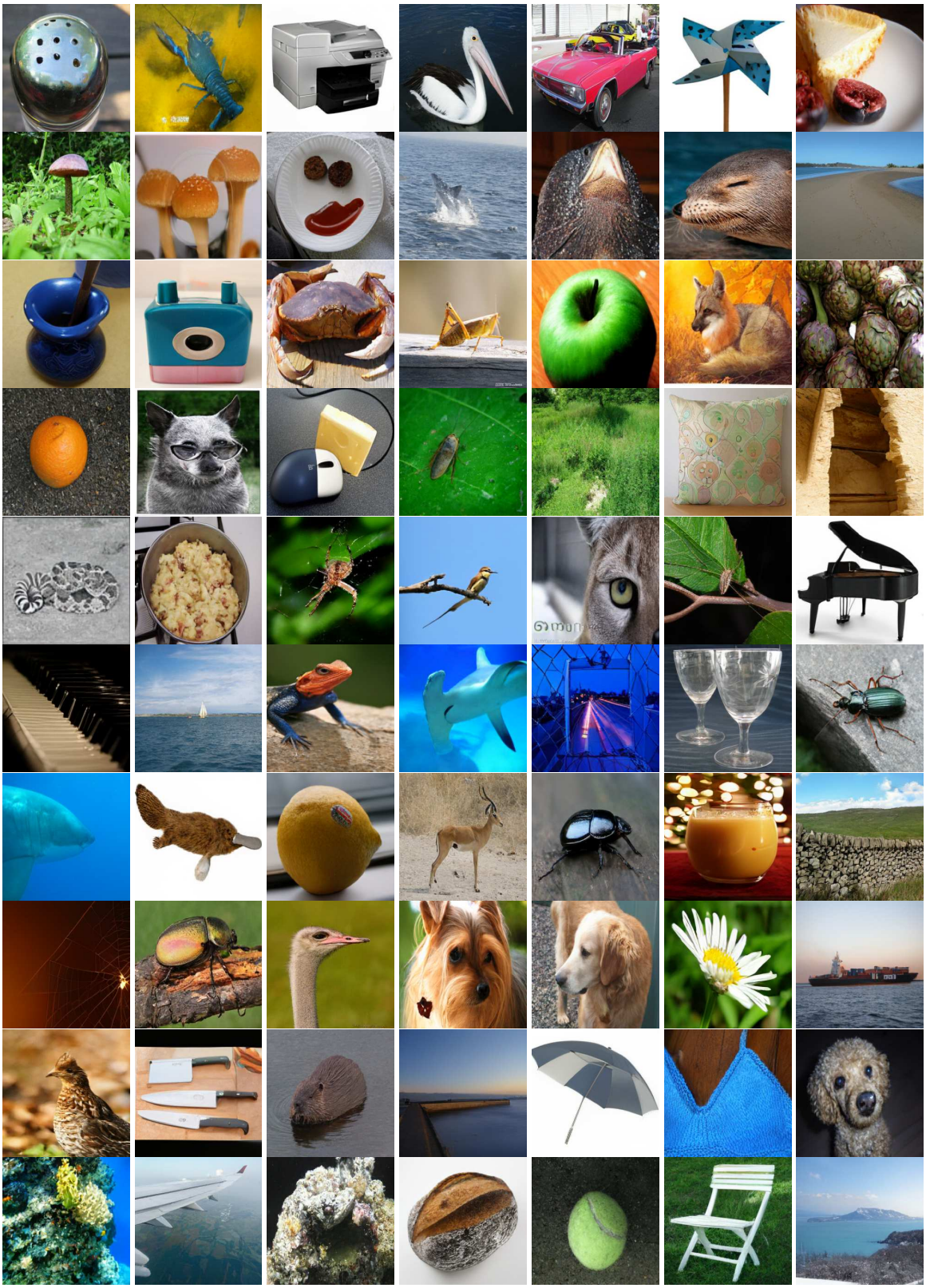

Figure F.5. Conditional generation examples of LEASE using ViT-B.

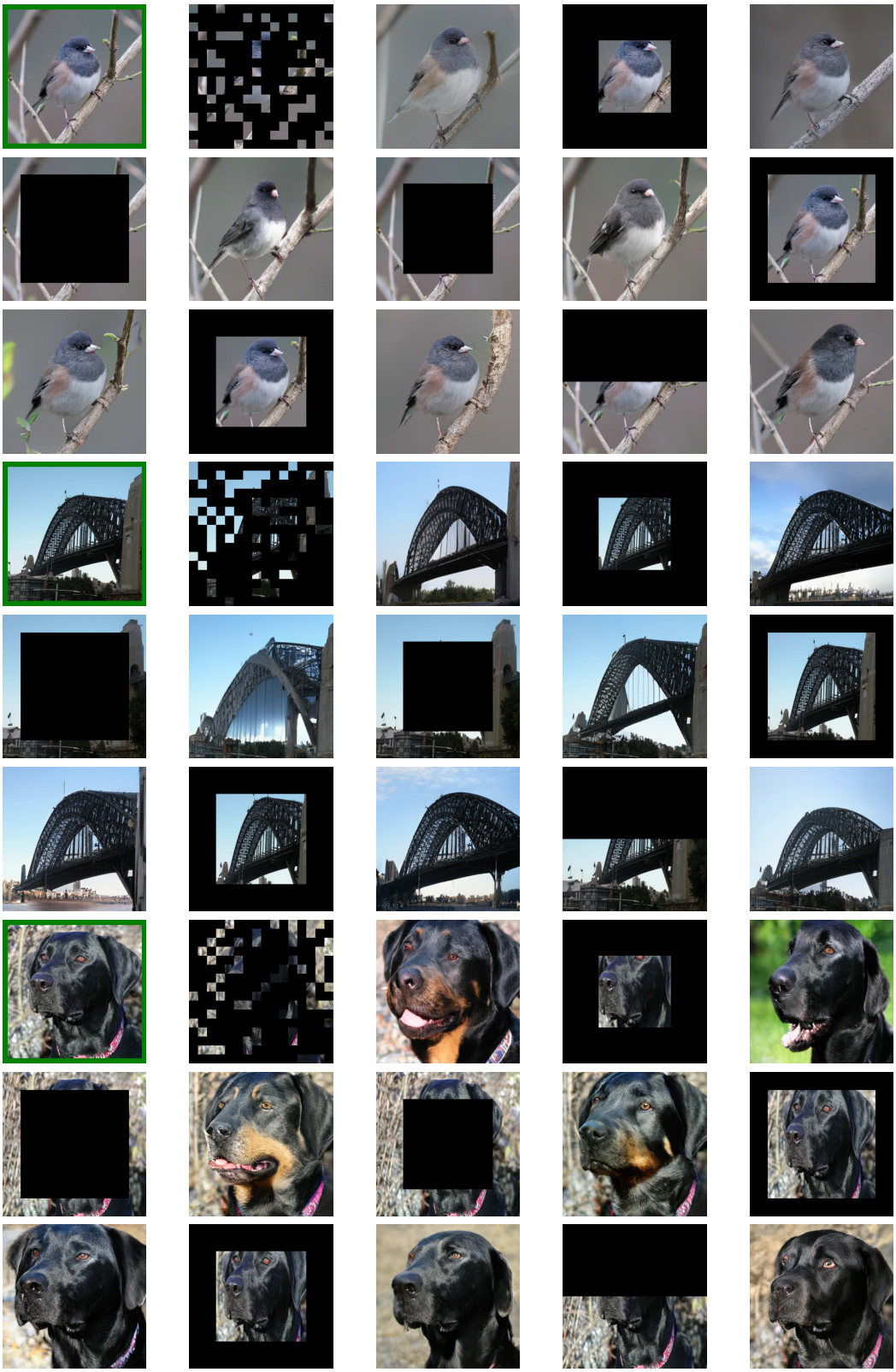

Figure F.6. Examples of image reconstruction, inpainting and outpainting for LEASE using ViT-B. Original image is marked in green.

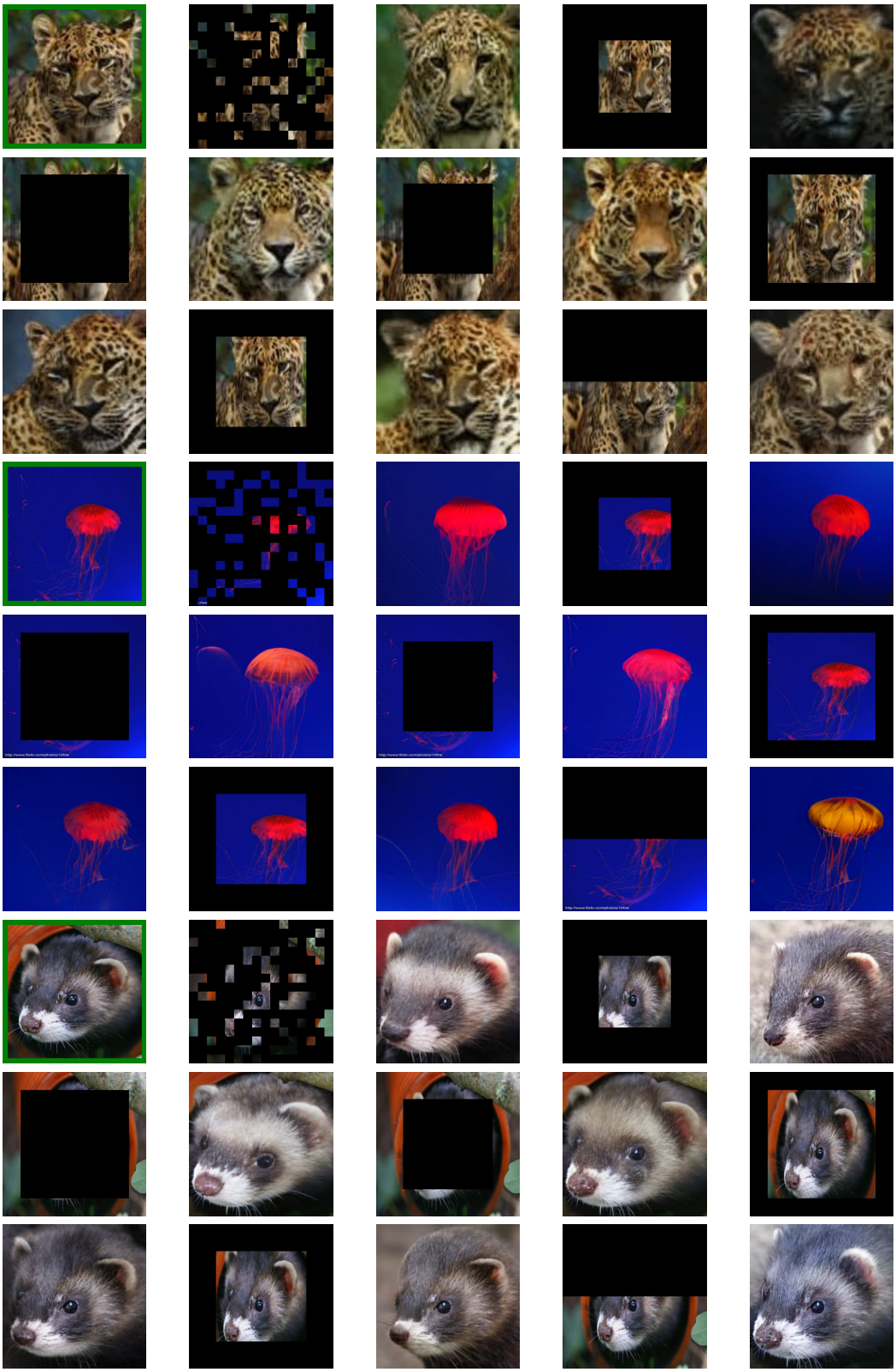

Figure F.7. Examples of image reconstruction, inpainting and outpainting for LEASE using ViT-B. Original image is marked in green.