Zero-Shot Textual Explanations via Translating Decision-Critical Features

Supplementary Material

A. Limitations

A.1. Quality of Concept Bank

Our method retrieves explanations from a pre-constructed concept bank; thus, the bank quality and coverage directly limit what can be explained. If relevant concepts are missing or under-represented, they cannot be retrieved at inference time, potentially yielding incomplete or biased explanations. This limitation is shared by many concept-based approaches, and improving concept-bank construction is an active research direction. TEXTER is agnostic to how the bank is built and can naturally benefit from improved construction methods as they emerge.

The number of descriptions per class is also a design choice that trades off coverage and noise. In practice, low retrieval confidence (e.g., uniformly low similarities or unstable rankings) can indicate insufficient coverage, suggesting that the concept bank should be expanded or adapted to the target domain.

A.2. Faithfulness evaluation

Following prior work, we mainly evaluate textual explanations with semantics-based metrics (Sec. 5.2.2). These metrics capture semantic consistency, but they do not directly measure causal faithfulness to the classifier's decision; a semantically plausible description may not reflect the evidence actually used by the model.

We also report a perturbation-based evaluation using CLIPSeg masks and class-probability changes (Appendix. C.3). However, this protocol assumes that evidence is well localizable and separable by masking, which can break for fine-grained or distributed cues. Moreover, when candidate descriptions share largely overlapping regions (e.g., *cat body* vs. *striped pattern*), the perturbation signal may be insensitive to the true decision basis.

Overall, robust and widely accepted faithfulness protocols for textual explanations remain underdeveloped, and establishing reliable metrics is an important direction for future work.

A.3. Computational overhead and dataset dependence

TEXTER introduces extra cost to train auxiliary modules (SAE/aligner). This training is offline and one-time (not per-image), but we do not claim efficiency; scalability to larger concept banks and higher-resolution settings remains a limitation.

TEXTER may also be dataset-dependent under distribution shift, as is common for post-hoc methods. If the concept bank or auxiliary modules are mismatched to the target domain, retrieval and concept-image validity can degrade, making explanations less reliable. We therefore use the validity check in Sec. 5.1 and treat low validity as a warning signal.

B. Implementation details

B.1. Details of SAE configuration

We adopt an overcomplete TopK Sparse Autoencoder [15], where the SAE embedding dimensionality exceeds that of the original feature space. Specifically, the embedding dimension (i.e., the dimensionality of $\Psi(f(x))$) is set to eight times that of the original feature vector $f(x)$, and the Top-K ratio is fixed at $10\,\%$, meaning that only the top $10\,\%$ entries with the largest magnitudes in each embedding vector are retained. The SAE is trained following Eq. (7) on the same dataset used to train each classifier (e.g., ImageNet) with batch size 1024, learning rate 5×10^{-4}, and the Adam optimizer for 10 epochs.

B.2. Details of concept bank construction

We use an LLM and a VLM to generate the concept bank $\mathcal{B}(x, c)$. Below, we describe the prompts used for each model.

The LLM is utilized to generate concepts that are generic to class c and not tied to a specific image. These concepts capture properties commonly associated with the class. Each concept description is constrained to a short phrase of 1–3 words to encourage compact, unit-like concepts, and descriptions that simply restate the class name are avoided. For the LLM input, we provide the target class name ({class_name}), the concepts already generated so far ({existing_concepts}), and an example question–answer pair, and obtain 10 new descriptions in a single inference. As a post-processing step, duplicate concepts are removed. Specifically, we use the following prompt:

```
Template variables (filled by the
    implementation as input variables):
- {class_name}: the target object
    class name.
- {existing_concepts}: already
    generated concepts.

Important guidelines for generating
    visual concepts:
1. Generate GENERAL concepts that can
```

```
    apply to many different photos of
    the same object type.
2. Include both OBJECT features (e.g.,
    shape, color, parts) AND CONTEXT
    features (e.g., background,
    environment, setting).
3. Keep concepts short and specific
    (1-3 words).
4. DO NOT include class names or
    object names directly.

Q: What are useful visual features for
    distinguishing a lemur in a photo?
A: There are several useful visual
    features to tell there is a lemur
    in a photo:
- long tail
- large eyes
- gray fur
- trees
- branches
- forest

Q: What are useful features for
    distinguishing a {class_name} in a
    photo?
Already generated concepts (DO NOT
    repeat these): {existing_concepts}.
A: There are several useful visual
    features to tell there is a
    {class_name} in a photo. Generate
    approximately 10 visual concepts
    to provide comprehensive coverage:
```

The VLM is employed to generate concepts that are grounded in the visual content of the image for class c. These concepts capture properties that are visually expressed in the specific input image and are therefore complementary to the generic class-level concepts generated by the LLM. As in the LLM setting, each concept description is constrained to a short phrase of 1–3 words, and descriptions that simply restate the class name are avoided. Unlike in the LLM setting, we additionally provide the image x as visual input to the VLM, along with the following text prompt. The generation procedure and post-processing (e.g., removal of duplicates) are kept consistent with the LLM case.

```
Template variables (filled by the
    implementation as input variables):
- {class_name}: the target object
    class name.
- {existing_concepts}: already
    generated concepts.

Important guidelines for generating
    visual concepts:
1. Generate DETAILED and SPECIFIC
    concepts that can apply to this
    image.
2. Include both OBJECT features (e.g.,
    shape, color, parts) AND CONTEXT
    features (e.g., background,
    environment, setting).
3. Keep concepts short and specific
    (1-3 words).
4. DO NOT include class names or
    object names directly.

Examples:
Q: Look at this image carefully. Based
    on what you can actually see in
    the image, identify useful visual
    features that help distinguish
    this as a koi fish.
A: There are several useful visual
    features to tell there is a koi
    fish in a photo:
- bright orange scales
- curved tail fin
- spotted pattern
- long body
- pointed snout
- water surface

Q: Look at this image carefully. Based
    on what you can actually see in
    the image, identify useful visual
    features that help distinguish
    this as a {class_name}.
Already generated concepts (DO NOT
    repeat these): {existing_concepts}.
A: There are several useful visual
    features to tell there is a
    {class_name} in a photo. Generate
    approximately 10 visual concepts
    to provide comprehensive coverage:
```

B.3. Other implementation details

As described in Sec. 4.1.2, MACO [11] is used to generate concept images. The concept images are generated using 512 iterations following Eq. (4), and all other parameters follow the original paper [11]. Since MACO tends to produce images with repeated patterns, we mitigate the redundancy this may introduce in textual explanation generation by randomly cropping multiple patches from the original concept image. The side length of each patch is drawn uniformly between 25 % and 30 % of the original image size, and the crop center is sampled from a Gaussian around the

Table 3. Quantitative evaluation using semantic-based metrics on ImageNet-1K. Each metric is computed between the explanations generated by each method and the concept images generated by the proposed method with SAE. Best results in bold.

Model	Method	CLIP-Score ↑	LPIPS (A) ↓	LPIPS (S) ↓	FS ↑
ResNet-18	Random	0.2295	0.7590	0.7065	0.6440
	Text-To-Concept	0.2275	0.7594	0.7054	0.6517
	TEXTER	**0.2333**	**0.7584**	**0.7036**	**0.6539**
ResNet-50	Random	0.2306	0.7651	0.7037	0.6814
	Text-To-Concept	0.2309	0.7618	0.7026	0.6856
	TEXTER	**0.2376**	**0.7596**	**0.7020**	**0.6961**
DINO ResNet-50	Random	0.2321	0.7579	0.6944	0.5256
	Text-To-Concept	0.2328	0.7532	0.6907	0.5349
	TEXTER	**0.2430**	**0.7506**	**0.6882**	**0.5432**
ViT	Random	0.2262	0.7736	0.6897	0.2597
	Text-To-Concept	0.2256	0.7746	0.6891	0.2644
	TEXTER	**0.2315**	**0.7693**	**0.6846**	**0.2668**
DINO ViT-S/8	Random	0.2260	0.7576	0.6915	0.4194
	Text-To-Concept	0.2243	0.7543	0.6898	0.4246
	TEXTER	**0.2337**	**0.7541**	**0.6892**	**0.4275**

Table 4. Quantitative evaluation using semantic-based metrics on ImageNet-1K. Each metric is computed between the explanations generated by each method and the original input images. Best results in bold.

Model	Method	CLIP-Score ↑	LPIPS (A) ↓	LPIPS (S) ↓	FS ↑
ResNet-18	Random	0.3043	0.7298	0.6180	0.7221
	Text-To-Concept	**0.3135**	**0.7196**	**0.6099**	**0.7321**
	TEXTER	0.3077	0.7278	0.6191	0.7255
ResNet-50	Random	0.3079	0.7306	0.6231	0.7640
	Text-To-Concept	**0.3174**	**0.7239**	**0.6144**	**0.7732**
	TEXTER	0.3108	0.7302	0.6219	0.7683
DINO ResNet-50	Random	0.3066	0.7330	0.6230	0.6284
	Text-To-Concept	**0.3178**	**0.7227**	**0.6139**	**0.6442**
	TEXTER	0.3098	0.7270	0.6210	0.6316
ViT	Random	0.3080	0.7306	0.6219	0.5113
	Text-To-Concept	**0.3143**	**0.7275**	**0.6160**	**0.5241**
	TEXTER	0.3113	0.7318	0.6237	0.5157
DINO ViT-S/8	Random	0.3081	0.7290	0.6185	0.6110
	Text-To-Concept	**0.3184**	**0.7215**	**0.6124**	**0.6363**
	TEXTER	0.3098	0.7307	0.6211	0.6140

image center and clipped to remain within the image boundaries. Each cropped patch is then resized to the original input resolution and mapped into the CLIP feature space via the aligner. We generate six patches by this process.

For textual explanation generation, the similarity between each aligned feature (derived from the six cropped patches) and every candidate in the concept bank $\mathcal{B}(x, c)$ is computed. For each candidate, the six similarity scores are averaged, and the candidates are sorted according to this averaged score to produce the final ranked list of textual explanations.

C. Additional quantitative results

C.1. Experimental settings for semantic-based metrics

We detail each metric introduced in Sec. 5.2.2.

For CLIP-Score, we use the ViT-B/16 image encoder. The text prompt is set as "a photo of {class_name} showing T," where {class_name} is the target class name and T is the set of generated concepts. Here, each description in T is separated by a comma.

For computing LPIPS and Feature Similarity, we generate an image x_g from T using Stable Diffusion. We use the "stable-diffusion-v1-5" model. The prompt for Stable Diffusion is formulated similarly to that used for CLIP-Score, as "a {class_name} showing T."

C.2. Additional results of semantic-based evaluations

We conduct quantitative evaluations using the semantic-based metrics introduced in Sec. 5.2.2 on the ImageNet-1K dataset. Each metric is computed on 1,000 randomly selected images, consisting of 200 classes with five images per class.

Table 3 shows the results comparing the concept images generated by the proposed method with the textual explanations produced by each method. Across all metrics and models, the proposed method consistently outperforms the baselines, indicating that it generates explanations with better semantic alignment to the concept images. Since ImageNet images typically contain a single large object, the type of visual features are relatively limited; therefore, the numerical differences appear smaller compared with Tab. 2 (evaluated on PASCAL VOC). Nevertheless, the proposed method consistently achieves superior scores, demonstrating its effectiveness.

As a complementary analysis, we also evaluate each method by comparing its explanations with the original input images, following prior work [46]. In this setting, higher scores indicate that the explanations describe the overall visual content of the input image well, rather than strictly reflecting the model's decision rationale. The results are summarized in Tab. 4. As expected, Text-To-Concept achieves the best performance across all models, which is consistent with its design: it aligns global image features with text and is intended to describe the input image itself. These results therefore complement Tabs. 2 and 3: while Text-To-Concept is better aligned with the input images, the proposed method provides explanations that are more semantically aligned with the decision-critical concept images.

Table 5. Evaluation using segmentation masks under insertion (Ins) and deletion (Del) settings. Best results are shown in bold.

Model	Method	Ins ↑	Del ↓
ResNet-18	Random	0.4690	0.2775
	Text-To-Concept	0.4756	0.2741
	TEXTER	**0.5081**	**0.2462**
ResNet-50	Random	0.4822	0.2999
	Text-To-Concept	0.4810	0.3080
	TEXTER	**0.4901**	**0.2797**
DINO ResNet-50	Random	0.5077	0.3383
	Text-To-Concept	0.4977	0.3380
	TEXTER	**0.5494**	**0.2880**
ViT	Random	0.3839	0.2916
	Text-To-Concept	0.3896	0.2922
	TEXTER	**0.4052**	**0.2699**
DINO ViT-S/8	Random	0.4319	0.3194
	Text-To-Concept	0.4465	0.3276
	TEXTER	**0.4698**	**0.2925**

C.3. Perturbation-based faithfulness evaluation

To complement semantics-based metrics, we additionally evaluate faithfulness with a perturbation test that links each retrieved text explanation to an image region. For each explanation, we use CLIPSeg [29] to obtain a segmentation mask scores, then perform insertion/deletion by progressively inserting high-score regions into a blank image or deleting them from the original image. We measure the area under the curve (AUC) of the target-class score change, following standard perturbation-based faithfulness protocols. We run this evaluation under the same setting as Sec. 5.2.

Table 5 shows the results. TEXTER yields consistently higher AUC than baselines, indicating that the regions implied by its retrieved explanations have a larger impact on the target prediction.

D. Additional qualitative results

D.1. Additional comparison results across models

We show additional comparisons of the generated explanations for the same prediction across models in Figs. F to H. These examples provide insight into each model's reasoning.

One example is the difference in the features recognized by CNNs and Transformers. In particular, Transformers use contextual information in addition to appearance features. For instance, in the *goldfish* example in Fig. F, the CNNs (ResNet-18 and ResNet-50) focus on specific colors such as "bright orange coloration," whereas the Transformer model DINO ViT-S/8 focuses on background cues such as "glass

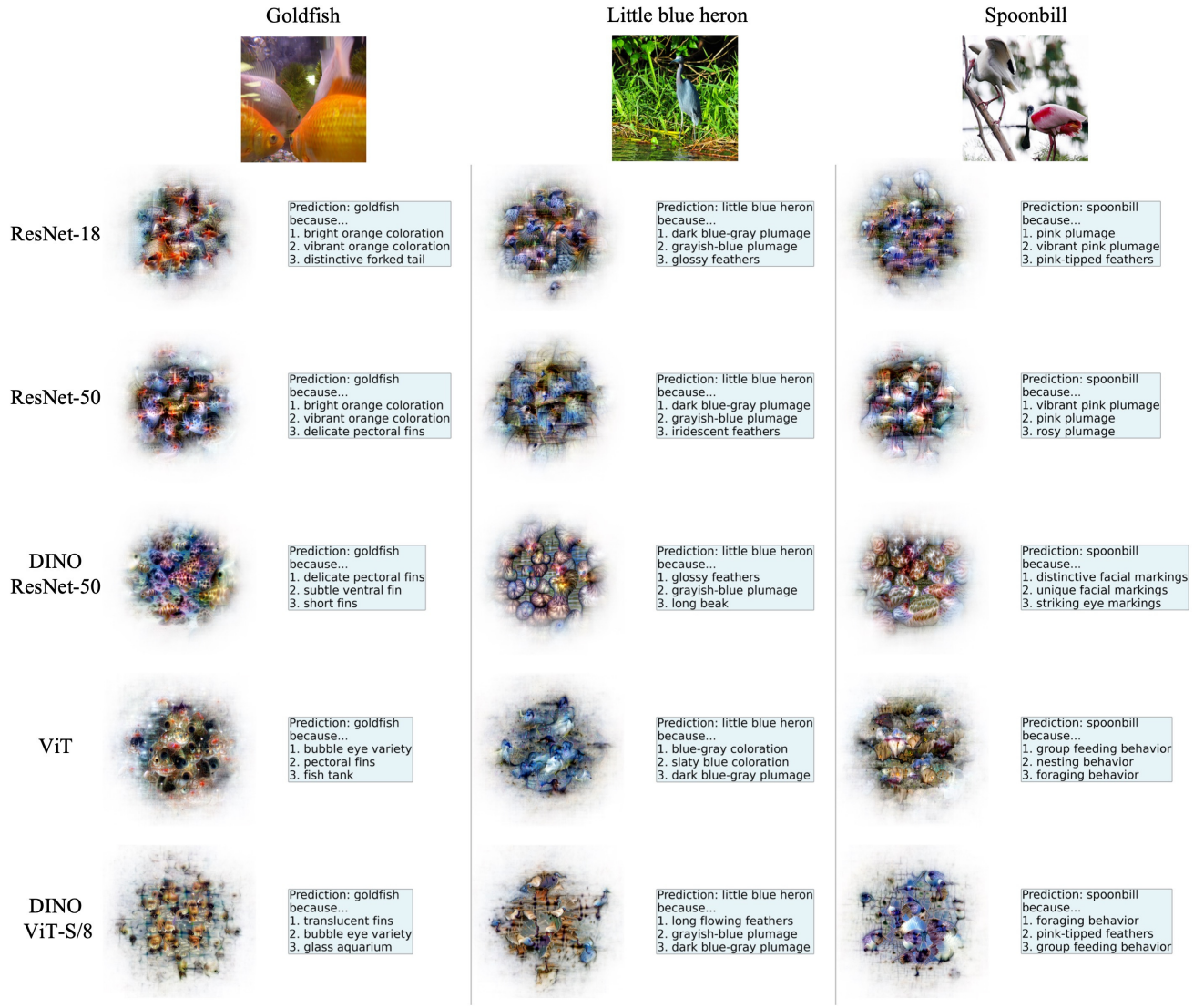

Figure F. Qualitative comparison of the textual explanations and concept images generated by the proposed method for the same prediction across ResNet-18, ResNet-50, DINO ResNet-50, ViT, and DINO ViT-S/8. Each row corresponds to the results from one model, and each column corresponds to one input image and its predicted class.

aquarium." Another example is *spoonbill* in Fig. F, where the Transformer models (ViT and DINO ViT-S/8) capture interactions between two birds, such as "group feeding behavior."

We also observe differences in the features used to distinguish similar classes. For example, as shown for *little blue heron* and *spoonbill* in Fig. F, the models often rely on color cues: the explanations for *little blue heron* include blue-related features such as "dark blue-gray plumage," whereas those for *spoonbill* include pink-related features such as "pink plumage". Another example is *saharan horned viper* and *green mamba* in Fig. G. Here, the models appear to distinguish the two classes based on the appearance of the snakes' scales. The explanations for *saharan horned viper* include desert-related features, such as "sand-colored scales" in ResNet-50 and "camouflage pattern" in ViT, whereas those for *green mamba* include green-related features, such as "green scales" in ResNet-50 and "yellow-green coloration" in DINO ViT-S/8.

As shown in these examples, textual explanations reveal which semantic cues the models rely on. While such insights are hard to obtain from attribution methods that only highlight important regions as heat maps [30, 48, 57], the proposed method additionally links each textual explanation to a corresponding concept image, grounding the text in concrete visual patterns and facilitating a deeper under-

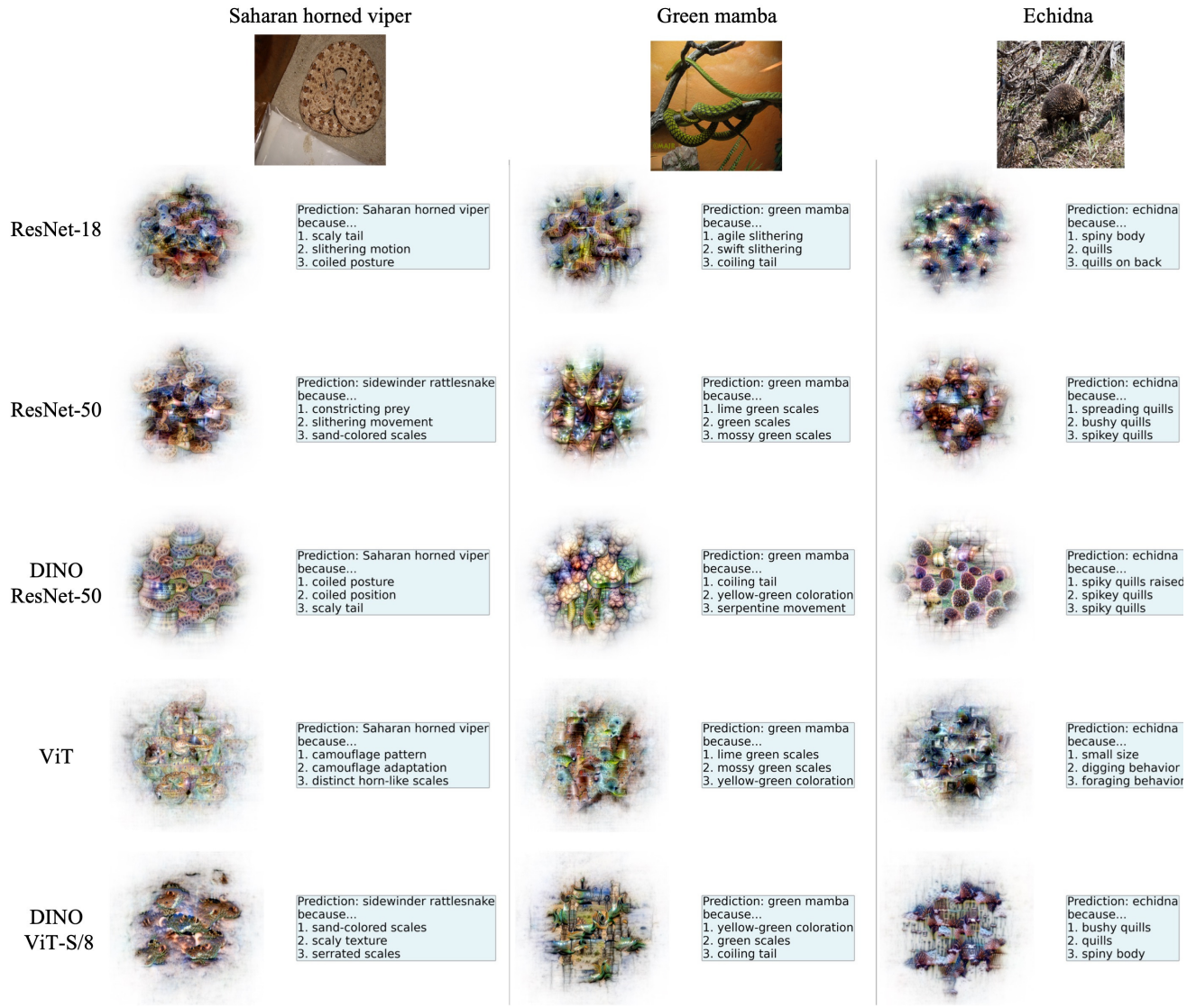

Figure G. Qualitative comparison of the textual explanations and concept images generated by the proposed method for the same prediction across ResNet-18, ResNet-50, DINO ResNet-50, ViT, and DINO ViT-S/8. Each row corresponds to the results from one model, and each column corresponds to one input image and its predicted class.

standing of the model's behavior.

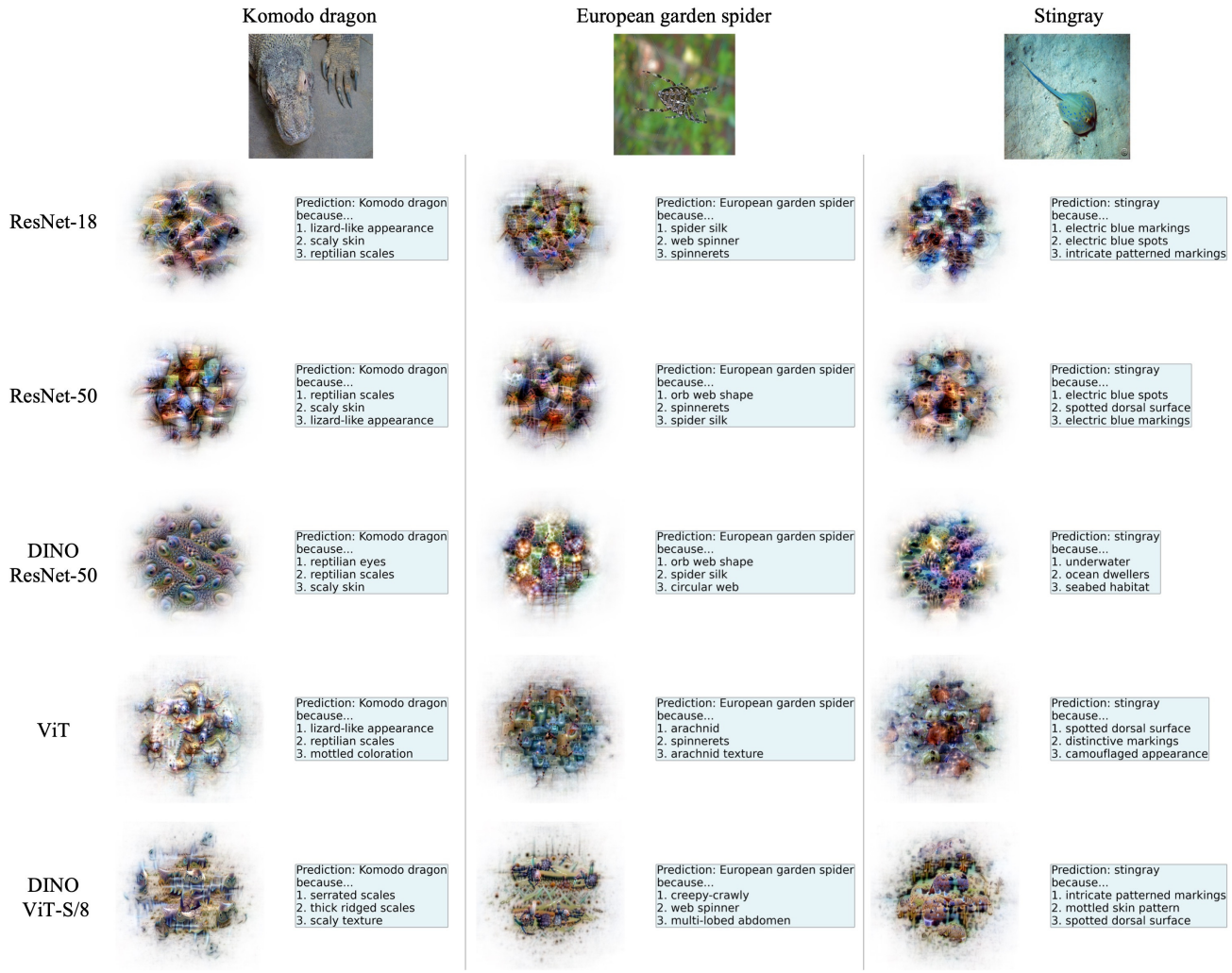

Figure H. Qualitative comparison of the textual explanations and concept images generated by the proposed method for the same prediction across ResNet-18, ResNet-50, DINO ResNet-50, ViT, and DINO ViT-S/8. Each row corresponds to the results from one model, and each column corresponds to one input image and its predicted class.