Supplementary Material for External Knowledge Injection for CLIP-Based Class-Incremental Learning

Da-Wei Zhou[1,2], Kai-Wen Li[1,2], Jingyi Ning[2], Han-Jia Ye[1,2](✉), Lijun Zhang[1,2], De-Chuan Zhan[1,2]
[1] School of Artificial Intelligence, Nanjing University
[2] National Key Laboratory for Novel Software Technology, Nanjing University
{zhoudw,likw,yehj,zhanglj,zhandc}@lamda.nju.edu.cn, ningjy@nju.edu.cn

In the main paper, we propose ExterNal knowledGe INjEction (ENGINE) for CLIP-based CIL. To enhance knowledge transfer from outside the dataset, we propose a dual-branch injection tuning framework that encodes informative knowledge from both visual and textual modalities. The visual branch is enhanced with data augmentation to enrich the visual features, while the textual branch leverages GPT-4 to rewrite discriminative descriptors. In addition to this on-the-fly knowledge injection, we also implement post-tuning knowledge by re-ranking the prediction results during inference. With the injected knowledge, the model can better capture informative features for downstream tasks as data evolves.

In this supplementary material, we provide more details about ENGINE, including more implementation details and experimental results.

- Section I introduces further analysis of ENGINE, including multiple runs, running time comparison, trainable parameter analysis, results of different backbones and different LLMs, and other baselines.
- Section II introduces the details of compared methods.
- Section III provides supplementary results of benchmark datasets to the main paper.
- Section IV provides more visualizations, including the visualization of prediction results and generated textual descriptions.

I. More results

This section includes more results on ENGINE, including the results with multiple runs using different random seeds, the details about trainable parameters, running time comparison, results with different pre-trained weights, and results with different LLMs to generate textual descriptions.

†Correspondence to: Han-Jia Ye (yehj@lamda.nju.edu.cn)

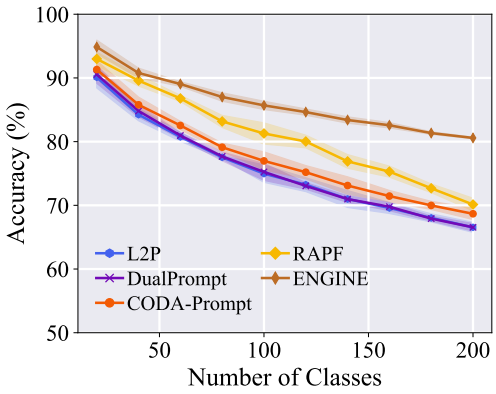


Figure 1. Results on ImageNet-R B0 Inc20 with multiple runs. ENGINE **consistently outperforms other methods by a substantial margin.**

I.1. Multiple Runs

In the main paper, the experimental results are conducted via splitting the classes with random seed 1993, which is a common practice in CIL [6]. To investigate the robustness of different methods, we also consider running the experiments multiple times using different random seeds. Specifically, we conduct the class split using random seed {1993,1994,1995,1996,1997}, and calculate the average performance and standard variance. We report the results in Figure 1 on ImageNet-R B0 Inc20.

As shown in the figure, ENGINE shows more robust results against other baselines. We find the results of ENGINE consistently outperform other competitors in multiple runs.

I.2. Trainable parameters

In this paper, we design ENGINE by extending knowledge injection unit per task. During inference, the injected features are aggregated as the final feature. As illustrated in Section 4.1 of the main paper, we can reparameterize these injection units by adding the weights since they are linear

Table 1. Number of trainable parameters on CIFAR100 B0 Inc10 setting.

Method	Trainable Parameters
L2P	161330
DualPrompt	333412
CODA-Prompt	3916900
RAPF	262144
ENGINE	524288

layers, *i.e.*, $\sum_p u_i^p$ and $\sum_p u_t^p$. Hence, the extra parameter size can be squeezed from $2 \times b \times d \times d$ to $2 \times d \times d$. We further report the number of trainable parameters in each compared method in Table 1. As we can infer from the table, ENGINE has the same scale of trainable parameters compared to other competitors, while having the best performance.

I.3. Running Time Comparison

In this section, we report the running time comparison of different methods. We utilize a single NVIDIA 4090 GPU to run the experiments and report the results in Figure 2. As we can infer from the figure, ENGINE requires less running time than CODA-Prompt and RAPF, while having the best performance. Experimental results verify the effectiveness of ENGINE.

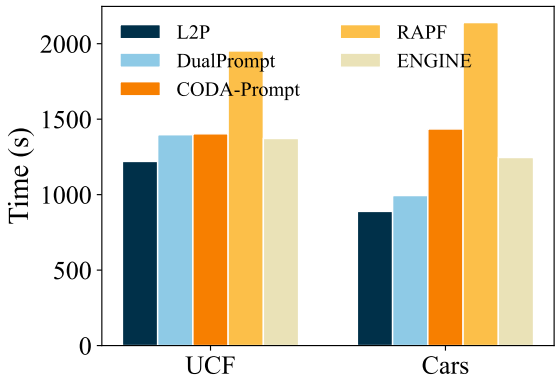


Figure 2. Running time comparison. ENGINE **has similar training time to other compared methods while having the best performance.**

Besides, the post-tuning process can be directly sped up with simple modifications. Specifically, we pre-compute the one-to-one text pairs and generate all corresponding text embeddings in advance. Assuming there are 100 classes and each pair contains n distinct descriptions, the total number of embeddings amounts to approximately $99 \times 100 \times n$. As a result, we only need to compute the descriptions in advance and subsequently retrieve the pre-calculated embeddings during inference, significantly reducing the computational overhead.

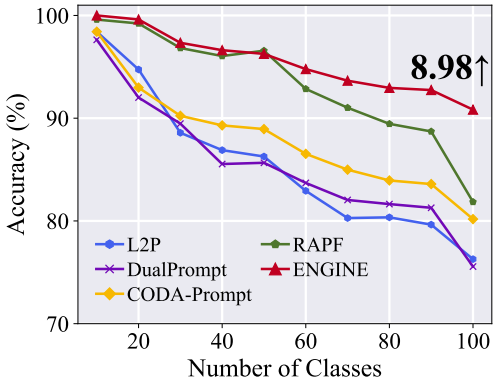


Figure 3. Experiments when using OpenAI weights on UCF B0 Inc10. ENGINE **consistently outperforms other methods with various backbone weights.**

I.4. Different backbones

In the main paper, we mainly consider CLIP with ViT-B/16 under LAION400M pre-trained weight [3][*] to conduct the experiments. In this section, we also provide the results with OpenAI pre-trained weight[†] on UCF B0 Inc10 in Figure 3. As we can infer from the figure, ENGINE consistently outperforms other methods with various backbone weights.

I.5. Different LLMs

In the main paper, we mainly utilize GPT-4o mini to generate the class descriptions. Since ENGINE is a general framework that is compatible with various LLMs, we consider Qwen2.5-72B [1] as the LLM to generate class descriptions, and conduct experiments on Food B0 Inc10. We report the comparison between GPT-4o mini and Qwen2.5-72B in Figure 4.

As shown in the figure, we find the performance results using different LLMs are quite similar, indicating the robustness of ENGINE when using different LLMs.

II. Introduction About Compared Methods

In this section, we introduce the details of the compared methods adopted in the main paper. For a fair comparison, all methods are based on the same pre-trained model. The details of the compared methods in Table 1 are listed as:

- **Finetune**: with a pre-trained CLIP as initialization, it finetunes CLIP for every new task. Hence, it suffers severe catastrophic forgetting on former tasks.

[*]https://github.com/mlfoundations/open_clip
[†]https://github.com/openai/CLIP

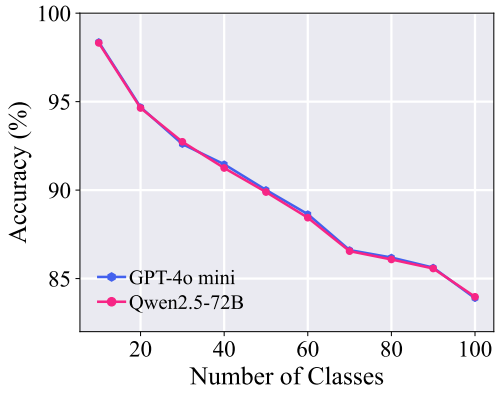


Figure 4. Results on Food B0 Inc10 with different LLMs for text description. ENGINE **is robust and compatible with various LLMs.**

- **CoOp [13]:** This approach freezes both the image encoder and text encoder of the pre-trained CLIP. It optimizes a learnable prompt tensor $\mathbf{t}$ using contrastive loss.
- **SimpleCIL [11]:** This method relies on the pre-trained image encoder and does not involve the text encoder. Hence, in the pre-trained CLIP, we drop the text branch and only use the visual branch for evaluation. The frozen image encoder extracts class centers (prototypes) for each new class, and a cosine classifier is utilized for classification. Since the model is not updated via backpropagation, it showcases the generalizability of the pre-trained vision encoder on downstream tasks.
- **ZS-CLIP [5]:** This baseline freezes the pre-trained CLIP and predicts the logits of each incoming class using cosine similarity. It serves as a reference for the performance of pre-trained CLIP on downstream tasks.
- **L2P [9]**: This method only involves the visual branch of CLIP. During model updating, it freezes the pre-trained weights and utilizes visual prompt tuning [4] to trace the new task's features. It builds instance-specific prompts with a prompt pool, which is constructed via key-value mapping.
- **DualPrompt [8]**: is an extension of L2P, which extends the prompt into two types, *i.e.*, general and expert prompts. The other details are kept the same with L2P, *i.e.*, using the prompt pool to build instance-specific prompts. This method only involves the visual branch of CLIP.
- **CODA-Prompt [7]**: noticing the drawback of instance-specific prompt selection, it aims to eliminate the prompt selection process by prompt reweighting. The prompt selection process is replaced with an attention-based prompt recombination. This method only involves the visual branch of CLIP.
- **RAPF [2]**: aims to enhance the continual learning ability of CLIP. It combines the hard class separation loss and decomposed parameter fusion to encode new knowledge into the CLIP model.

The above methods are exemplar-free, which do not require using exemplars. We also compare some exemplar-based methods in Table 2 of the main paper as follows:

- **iCaRL [6]**: This method only involves the visual branch of CLIP. It utilizes knowledge distillation and exemplar replay to recover former knowledge. It also utilizes the nearest center mean classifier for final classification.
- **MEMO [10]**: This method only involves the visual branch of CLIP. It decouples the network structure into specialized (deep) and generalized (shallow) layers and extends specialized layers based on the shared generalized layers. Hence, the memory cost for network expansion decreases from a whole backbone to generalized blocks. In the implementation, we follow [10] to decouple the vision transformer at the last transformer block.
- **PROOF [12]**: aims to enhance CLIP's continual learning ability by learning expandable projection layers and cross-modal fusion module. The prototypes of historical visual and textual features are passed through the cross-modal fusion for further matching.

In the experiments, we reimplement the above methods based on their source code and PyCIL[‡].

III. Full Results

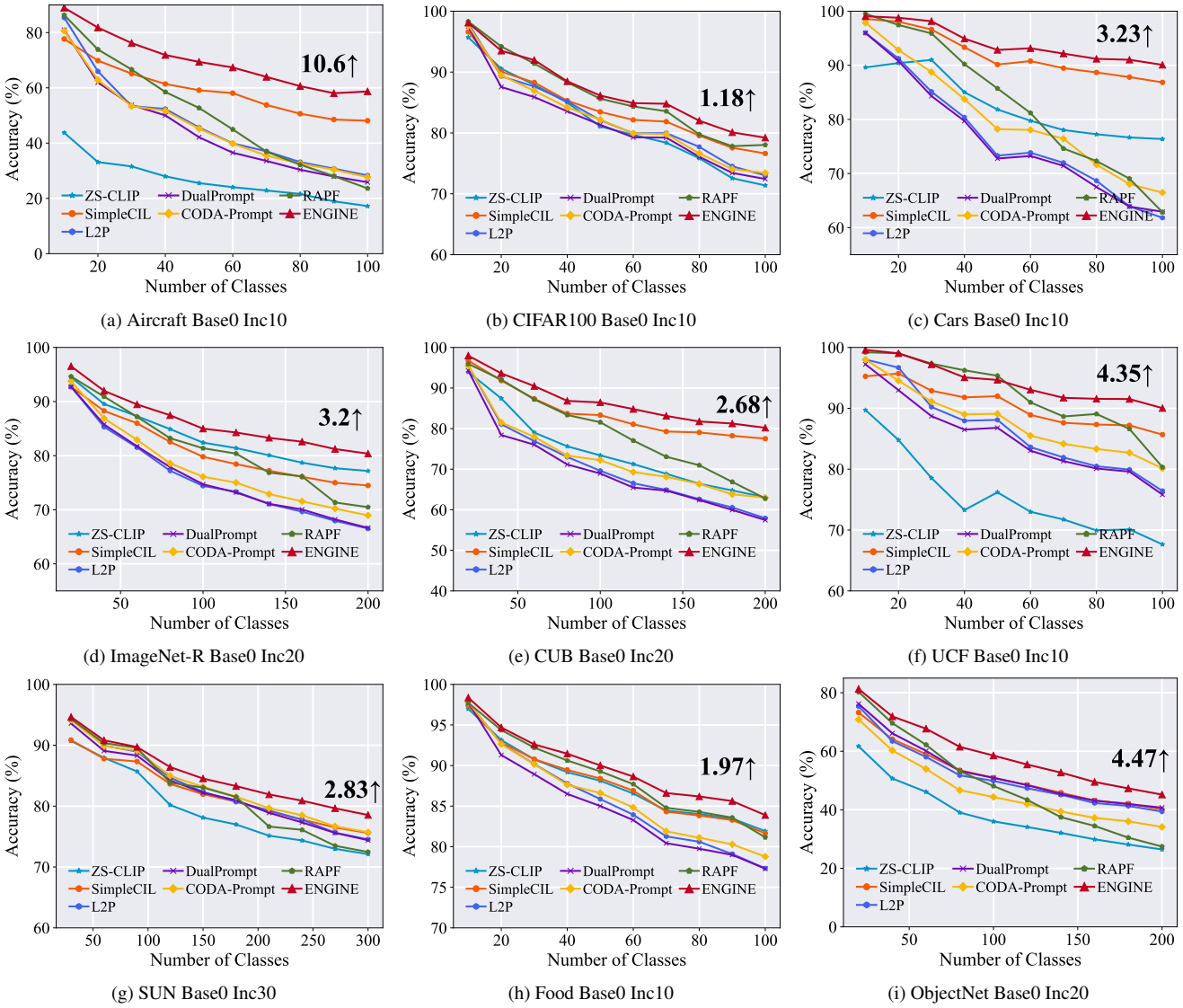
In this section, we show more experimental results of different methods. In the main paper, we only report three typical learning trends among compared methods. In this section, we report the full results corresponding to Table 1 of the main paper. Specifically, we report the incremental performance of different methods with 0 base classes in Figure 5 and half base classes in Figure 6. As shown in these results, ENGINE consistently outperforms other methods on different datasets and different data splits by a substantial margin.

IV. More Visualizations

IV.1. Adjusted predictions

In this section, we provide more visualizations of the effect of post-tuning knowledge injection in Figure 7. As shown in the figure, post-tuning efficiently adjusts the predictions of the model to highlight the ground truth class.

[‡]https://github.com/G-U-N/PyCIL



(a) Aircraft Base0 Inc10 (b) CIFAR100 Base0 Inc10 (c) Cars Base0 Inc10

(d) ImageNet-R Base0 Inc20 (e) CUB Base0 Inc20 (f) UCF Base0 Inc10

(g) SUN Base0 Inc30 (h) Food Base0 Inc10 (i) ObjectNet Base0 Inc20

Figure 5. Incremental performance of different methods on B0 setting. We report the performance gap after the last incremental stage of ENGINE and the runner-up method at the end of the line. All methods utilize the same CLIP pre-trained weight.

IV.2. Generated descriptions

In this section, we give examples of the generated descriptions of class-wise discriminative features. We give the examples on CIFAR100:

Camel versus cattle:

Q: What are unique visual features of camel compared to cattle in a photo? Focus on their key visual differences.
A: Camel: **1.** Tall, humped back which distinguishes it from cattle. **2.** Long legs and a proportionally longer neck, giving a different silhouette. **3.** Typically covered in coarse fur, often light brown or beige. **4.** Feet have padding for desert terrains, contrasting with cattle's hooves. **5.**Can often be seen with a distinctive gait, differing from the more sturdy stance of cattle.
Cattle: **1.** Large, bulky bodies with short legs. **2.** Varying coat colors from black to brown to white. **3.** Prominent horns on some breeds, depending on gender. **4.** Large, expressive eyes with long eyelashes. **5.** Commonly seen grazing in fields or pastures.

Shark versus ray:

Q: What are unique visual features of shark compared to ray in a photo? Focus on their key visual differences.

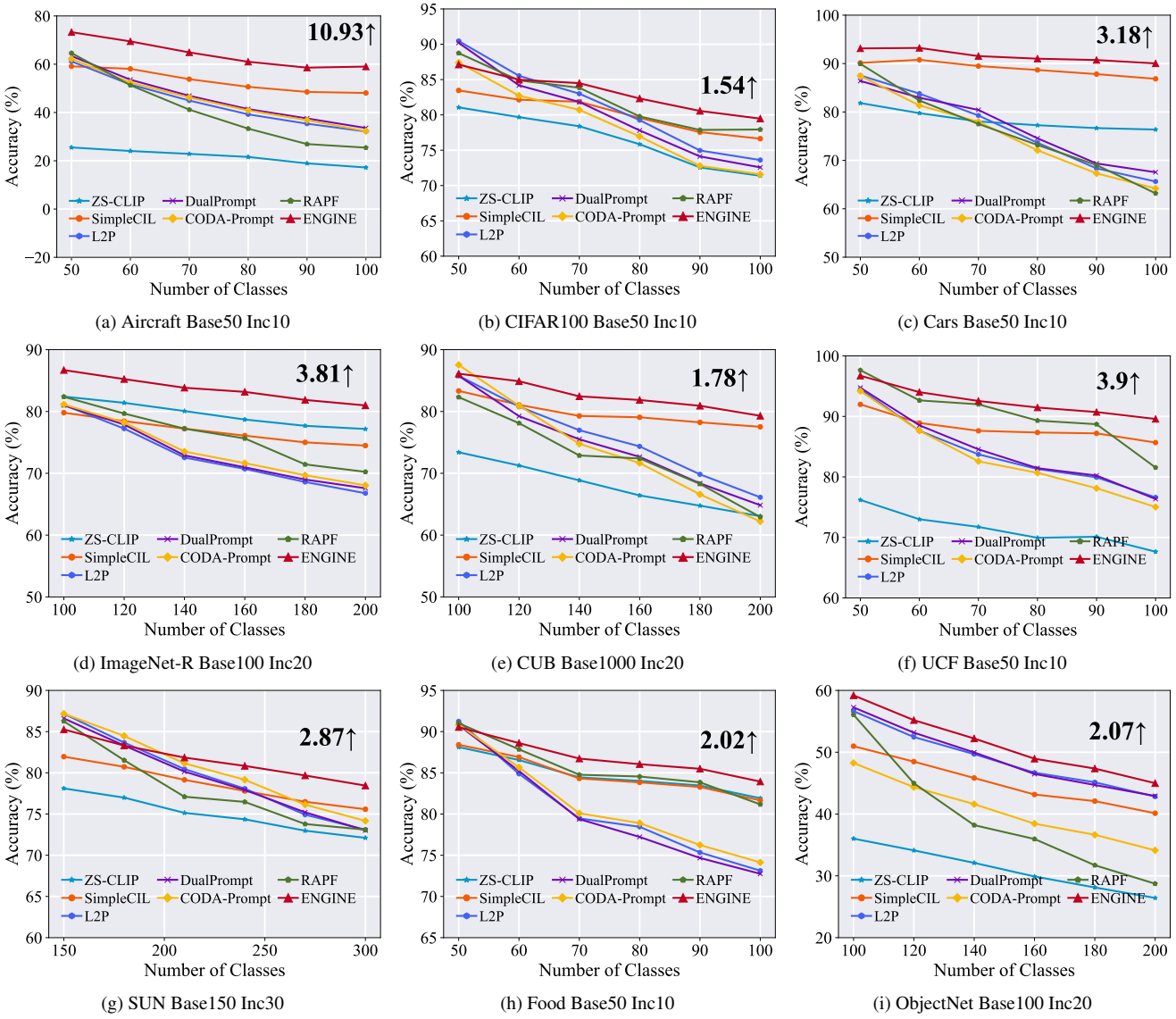


Figure 6. Incremental performance of different methods on half-base setting. We report the performance gap after the last incremental stage of ENGINE and the runner-up method at the end of the line. All methods utilize the same CLIP pre-trained weight.

A: Shark: **1.**Streamlined body with a pointed snout and sharp teeth. **2.** A dorsal fin prominently displayed on the top.**3.** Pelvic fins and pectoral fins that assist in maneuvering. **4.**Typically gray or blueish skin, often with a lighter belly. **5.** Can be shown swimming in a sleek, powerful posture.
Ray: **1.** Flattened body shape with a broad, disc-like structure. **2.** Long, pointed snout extending from the disc. **3.** Wing-like pectoral fins that extend outwards. **4.** Short tail that may have a spine or barb. **5.** Coloration often varying from gray to brown.

Bed versus couch:

Q: What are unique visual features of bed compared to couch in a photo? Focus on their key visual differences.
A: Bed: **1.**Typically rectangular shape, varying in size. **2.** Soft, cushioned surface with a mattress.**3.** Often has a headboard or footboard. **4.**May include bedding such as blankets and pillows. **5.** Primarily found indoors in a home setting.
Couch: **1.** Large, upholstered furniture piece with a soft seating area. **2.** Typically consists of cushions that vary in firmness. **3.** Often rectangular or L-shaped in design. **4.** Can have armrests and back support, with decorative fabric. **5.** Used in

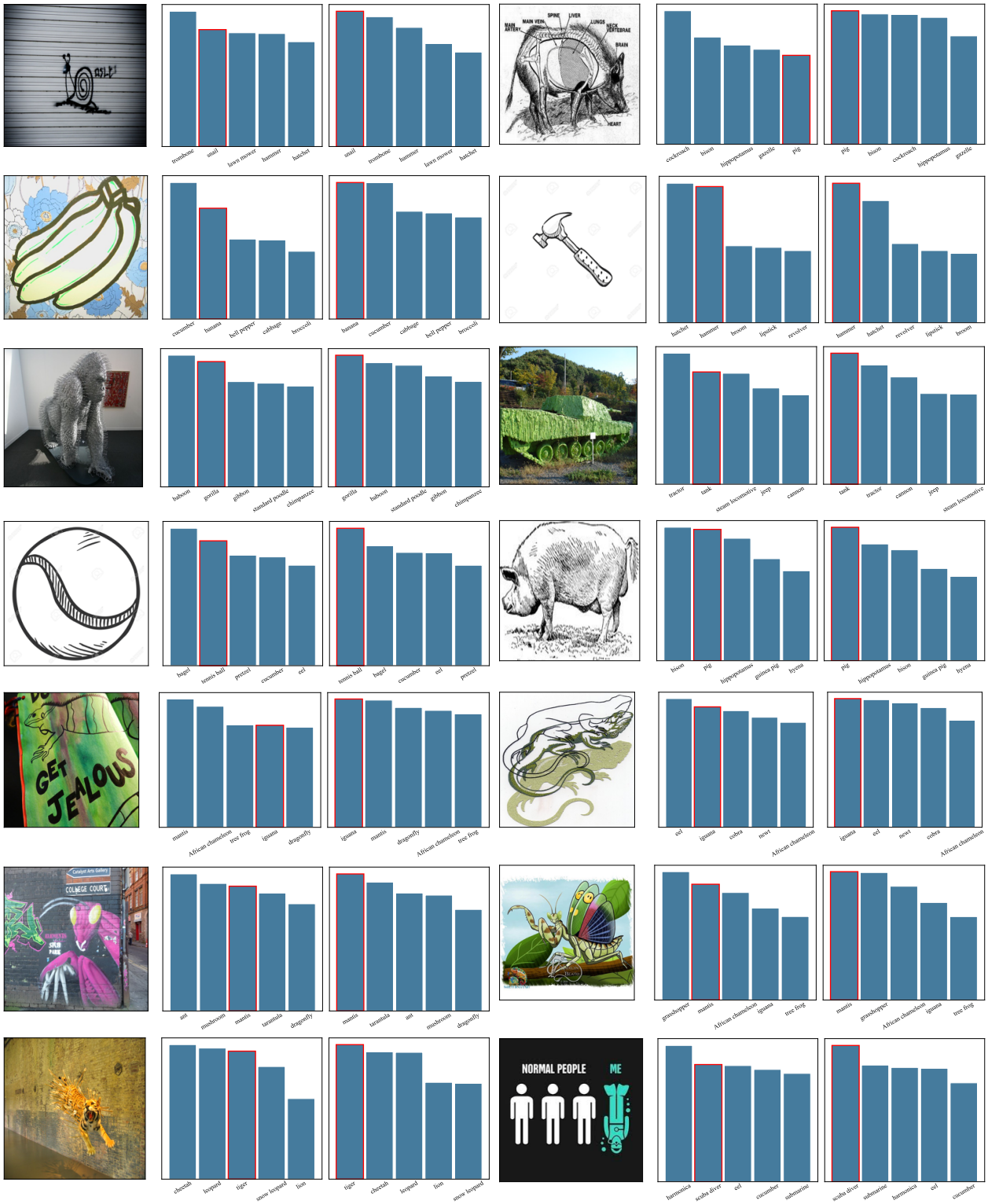


Figure 7. **Left:** Input. **Middle:** Top-5 predictions before post-tuning knowledge injection. **Right:** Top-5 predictions after post-tuning. More cases are shown in the supplementary.

living rooms or lounges, contrasting a flat surface.

We also give the examples on ImageNet-R:
Harp versus electric guitar:

Q: What are unique visual features of harp compared to electric guitar in a photo? Focus on their key visual differences.
A:Harp: **1.**Large, triangular frame with a curved top and vertical strings. **2.** Tall and elegant, often made of wood.**3.** Strings are arranged in a straight line from top to bottom, creating tension. **4.**Pedals at the base for changing pitch on certain types. **5.** Generally features ornate carvings and a polished finish.
Electric guitar: **1.** Solid body with a distinctive curvy or pointed shape. **2.** Strings visible on top along a long neck, attached to a circular body. **3.**Often has pickups, knobs, and frets, creating a complex visual profile. **4.** Typically made of wood with a shiny finish and various colors. **5.** Centered around sound production through electrical means unlike the strings of a harp that are different in construction.

Pizza versus burrito:

Q: What are unique visual features of pizza compared to burrito in a photo? Focus on their key visual differences.
A: Pizza: **1.**Flat, round shape with a raised crust along the perimeter. **2.** Visible layers of toppings, sauce, and cheese creating a colorful appearance.**3.** Often has a glossy sheen from melted cheese, unlike the matte finish of a burrito. **4.**Slices that can show a variety of ingredients and color contrasts. **5.** No wrapping it is served open-faced, showcasing toppings, unlike a burrito.
Burrito: **1.** Cylindrical shape wrapped in a soft, flat tortilla. **2.** Often showcases a colorful mixture of fillings (rice, beans, meat). **3.**Typically closed at both ends, contrasting with pizza open slices. **4.** Appears larger and thicker than a typical slice of pizza. **5.** Topping textures hidden within, rather than spread across the surface.

References

[1] Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, Binyuan Hui, Luo Ji, Mei Li, Junyang Lin, Runji Lin, Dayiheng Liu, Gao Liu, Chengqiang Lu, Keming Lu, Jianxin Ma, Rui Men, Xingzhang Ren, Xuancheng Ren, Chuanqi Tan, Sinan Tan, Jianhong Tu, Peng Wang, Shijie Wang, Wei Wang, Shengguang Wu, Benfeng Xu, Jin Xu, An Yang, Hao Yang, Jian Yang, Shusheng Yang, Yang Yao, Bowen Yu, Hongyi Yuan, Zheng Yuan, Jianwei Zhang, Xingxuan Zhang, Yichang Zhang, Zhenru Zhang, Chang Zhou, Jingren Zhou, Xiaohuan Zhou, and Tianhang Zhu. Qwen technical report. *arXiv preprint arXiv:2309.16609*, 2023. 2

[2] Linlan Huang, Xusheng Cao, Haori Lu, and Xialei Liu. Class-incremental learning with clip: Adaptive representation adjustment and parameter fusion. In *ECCV*, pages 214–231, 2024. 3

[3] Gabriel Ilharco, Mitchell Wortsman, Ross Wightman, Cade Gordon, Nicholas Carlini, Rohan Taori, Achal Dave, Vaishaal Shankar, Hongseok Namkoong, John Miller, Hannaneh Hajishirzi, Ali Farhadi, and Ludwig Schmidt. Openclip, 2021. 2

[4] Menglin Jia, Luming Tang, Bor-Chun Chen, Claire Cardie, Serge J. Belongie, Bharath Hariharan, and Ser-Nam Lim. Visual prompt tuning. In *ECCV*, pages 709–727, 2022. 3

[5] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *ICML*, pages 8748–8763, 2021. 3

[6] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *CVPR*, pages 2001–2010, 2017. 1, 3

[7] James Seale Smith, Leonid Karlinsky, Vyshnavi Gutta, Paola Cascante-Bonilla, Donghyun Kim, Assaf Arbelle, Rameswar Panda, Rogerio Feris, and Zsolt Kira. Coda-prompt: Continual decomposed attention-based prompting for rehearsal-free continual learning. In *CVPR*, pages 11909–11919, 2023. 3

[8] Zifeng Wang, Zizhao Zhang, Sayna Ebrahimi, Ruoxi Sun, Han Zhang, Chen-Yu Lee, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer Dy, et al. Dualprompt: Complementary prompting for rehearsal-free continual learning. In *ECCV*, pages 631–648, 2022. 3

[9] Zifeng Wang, Zizhao Zhang, Chen-Yu Lee, Han Zhang, Ruoxi Sun, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer Dy, and Tomas Pfister. Learning to prompt for continual learning. In *CVPR*, pages 139–149, 2022. 3

[10] Da-Wei Zhou, Qi-Wei Wang, Han-Jia Ye, and De-Chuan Zhan. A model or 603 exemplars: Towards memory-efficient class-incremental learning. In *ICLR*, 2023. 3

[11] Da-Wei Zhou, Zi-Wen Cai, Han-Jia Ye, De-Chuan Zhan, and Ziwei Liu. Revisiting class-incremental learning with pre-trained models: Generalizability and adaptivity are all you need. *International Journal of Computer Vision*, 133:1012–1032, 2025. 3

[12] Da-Wei Zhou, Yuanhan Zhang, Yan Wang, Jingyi Ning, Han-Jia Ye, De-Chuan Zhan, and Ziwei Liu. Learning without forgetting for vision-language models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 47(6): 4489–4504, 2025. 3

[13] Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. Learning to prompt for vision-language models. *IJCV*, 130(9):2337–2348, 2022. 3