Supplementary Material for "Defense-Prefix for Preventing Typographic Attacks on CLIP"

Hiroki Azuma Yusuke Matsui
The University of Tokyo, Japan
{azuma, matsui}@hal.t.u-tokyo.ac.jp

A. Prompts

In Sec. 3.2, we use templates to prepare input text t_i and t_i^{DP}. For training, we randomly choose a template from hand-crafted prompts in each iteration. For hand-crafted, we use 81 prompts: ('<CLS>.', 'a photo of a <CLS>.', 'a bad photo of a <CLS>.', 'a photo of many <CLS>.', 'a sculpture of a <CLS>.', 'a photo of the hard to see <CLS>.', 'a low resolution photo of the <CLS>.', 'a rendering of a <CLS>.', 'graffiti of a <CLS>.', 'a bad photo of the <CLS>.', 'a cropped photo of the <CLS>.', 'a tattoo of a <CLS>.', 'the embroidered <CLS>.', 'a photo of a hard to see <CLS>.', 'a bright photo of a <CLS>.', 'a photo of a clean <CLS>.', 'a photo of a dirty <CLS>.', 'a dark photo of the <CLS>.', 'a drawing of a <CLS>.', 'a photo of my <CLS>.', 'the plastic <CLS>.', 'a photo of the cool <CLS>.', 'a close-up photo of a <CLS>.', 'a black and white photo of the <CLS>.', 'a painting of the <CLS>.', 'a painting of a <CLS>.', 'a pixelated photo of the <CLS>.', 'a sculpture of the <CLS>.', 'a bright photo of the <CLS>.', 'a cropped photo of a <CLS>.', 'a plastic <CLS>.', 'a photo of the dirty <CLS>.', 'a jpeg corrupted photo of a <CLS>.', 'a blurry photo of the <CLS>.', 'a photo of the <CLS>.', 'a good photo of the <CLS>.', 'a rendering of the <CLS>.', 'a <CLS> in a video game.', 'a photo of one <CLS>.', 'a doodle of a <CLS>.', 'a close-up photo of the <CLS>.', 'the origami <CLS>.', 'the <CLS> in a video game.', 'a sketch of a <CLS>.', 'a doodle of the <CLS>.', 'a origami <CLS>.', 'a low resolution photo of a <CLS>.', 'the toy <CLS>.', 'a rendition of the <CLS>.', 'a photo of the clean <CLS>.', 'a photo of a large <CLS>.', 'a rendition of a <CLS>.', 'a photo of a nice <CLS>.', 'a photo of a weird <CLS>.', 'a blurry photo of a <CLS>.', 'a cartoon <CLS>.', 'art of a <CLS>.', 'a sketch of the <CLS>.', 'a embroidered <CLS>.', 'a pixelated photo of a <CLS>.', 'itap of the <CLS>.', 'a jpeg corrupted photo of the <CLS>.', 'a good photo of a <CLS>.', 'a plushie <CLS>.', 'a photo of the nice <CLS>.', 'a photo of the small <CLS>.', 'a photo of the weird <CLS>.', 'the cartoon <CLS>.', 'art of the <CLS>.', 'a drawing of the <CLS>.', 'a photo of the large <CLS>.', 'a black and white photo of a <CLS>.', 'the plushie <CLS>.', 'a dark photo of a <CLS>.', 'itap of a <CLS>.', 'graffiti of the <CLS>.', 'a toy <CLS>.', 'itap of my <CLS>.', 'a photo of a cool <CLS>.', 'a photo of a small <CLS>.', 'a tattoo of the <CLS>.',)

In Sec. 4.2, we evaluate our method through classification. For classification, we use hand-crafted prompts (Table A).

B. Synthetic typographic attack datasets

In this Sec., we will explain the details of the training data in Sec. 3.2 and test data in Sec. 4.2. When we train the DP vector (Sec. 3.2) and conduct experiments on classification (Sec. 4.2), we use synthetic typographic attack datasets. For training data, we add text to images from ImageNet-100 (Figure A). For test data, we add text to images from ten classification datasets (Figure B): ImageNet [4], Caltech101 [5], OxfordPets [14], StanfordCars [9], Flowers102 [13], Food101 [1], FGVCAircraft [11], DTD [2], SUN397 [16], EuroSAT [7]. To make typographic attack datasets, we followed the way of PAINT [8]. We resize the short dimension to 224 pixels using bicubic interpolation and crop 224 pixels by 224 pixels in the center, which is the standard CLIP [15] resize and crop augmentation. For fonts, we randomly choose from three fonts: Roman, Courier, Times. For font size, we randomly sample between 20 and 40 points. Also, we randomize over eight colors: red, green, blue, cyan, magenta, yellow, white, and black. We outline text with a 1-point shadow that is a different color from the main font color. The text is randomly placed in the image such that whole words are visible. Text is chosen from the class labels of the dataset except for the correct image labels.

For object detection, we also make synthetic typographic attack datasets using COCO [10] and LVIS [6] (Figure C). We use AdobeVFPrototype as a font. We randomize over eight colors: red, green, blue, cyan, magenta, yellow, white, and black. We outline text with a 1-point shadow that is a different color from the main font color. The text is randomly placed in each

Table A. **Prompts used for inference**

Dataset	Prompt
ImageNet	"a photo of a <CLS>."
Caltech101	"a photo of a <CLS>."
OxfordPets	"a photo of a <CLS>, a type of pet."
StanfordCars	"a photo of a <CLS>."
Flowers102	"a photo of a <CLS>, a type of flower."
Food101	"a photo of a <CLS>, a type of food."
FGVCAircraft	"a photo of a <CLS>, a type of aircraft."
DTD	"<CLS> texture."
SUN397	"a photo of a <CLS>."
EuroSAT	"a centered satellite photo of a <CLS>."
Real-world typographic attack datasets	"a photo of a <CLS>."

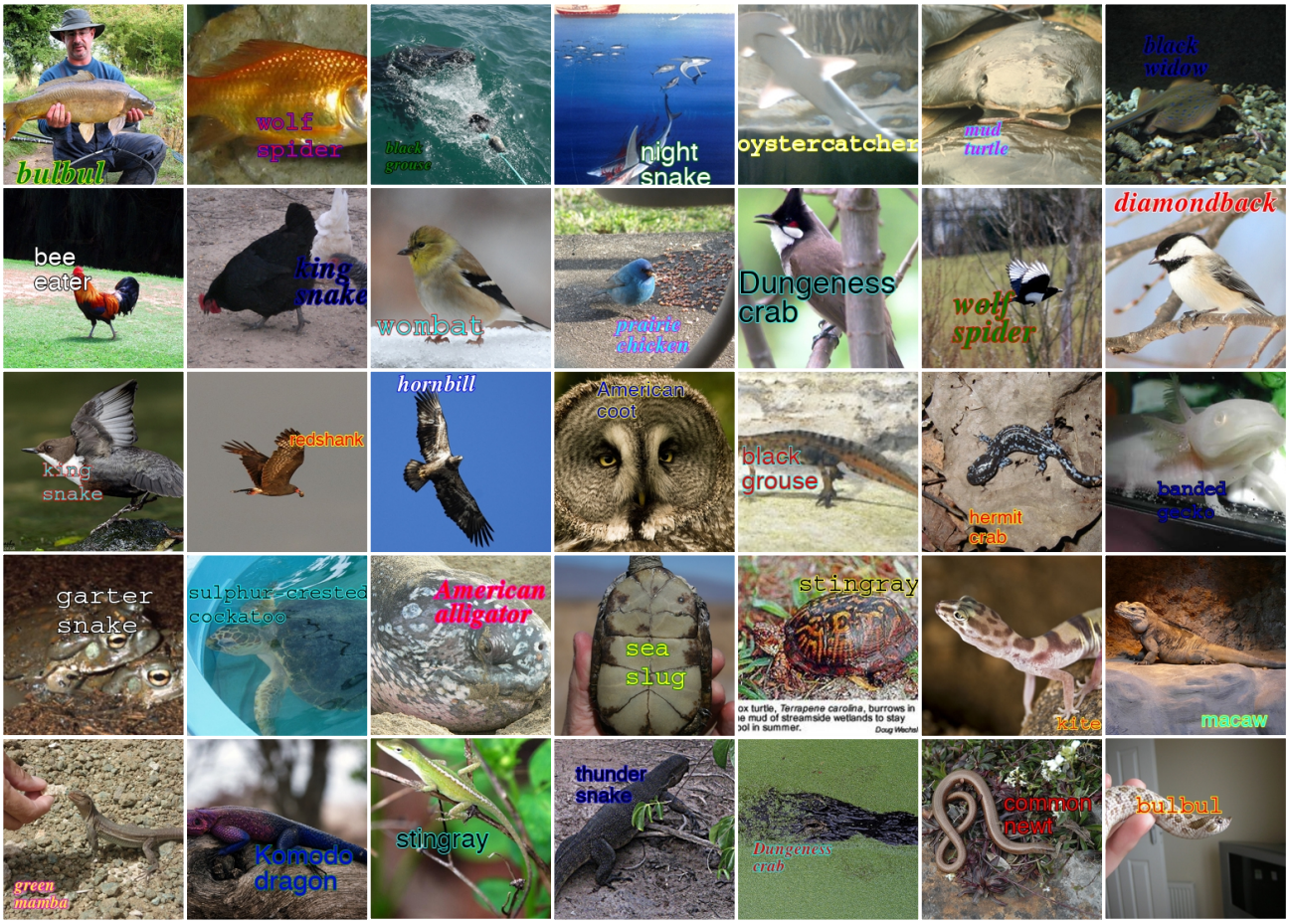

Figure A. **Images sampled from our training dataset.** The dataset consists of images from ImageNet-100 with synthesized text.

bounding box such that the whole words are visible. We adjust the font size so that the width of the text is less than 0.8 times the width of the bounding box.

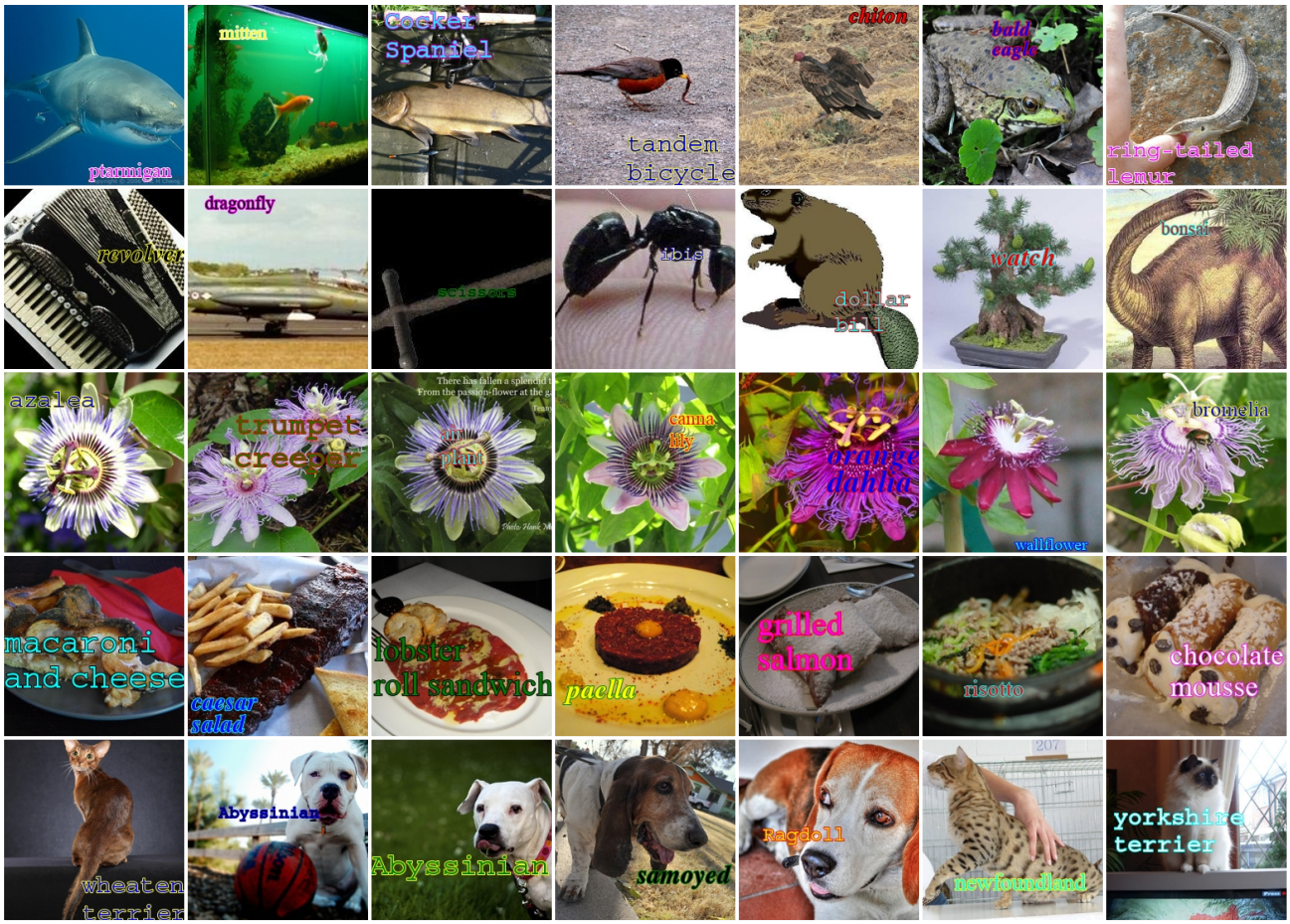

Figure B. **Images sampled from our test datasets.** We use ten datasets to make test data for synthetic typographic attack datasets.

C. RTA-100

In Sec. 4.2, we use real-world typographic attack datasets. To increase the test data, we take pictures and make RTA-100, which is the biggest real-world typographic attack dataset (Figure D). We put tags that are labeled incorrect classes to objects. We choose the incorrect labels of the tags from the objects in our dataset. We take pictures from 10cm to 2m from the objects such that whole words are visible. For example, we write "pen" on the tag and put it on a frisbee. Then, we take a photo of the object. For fonts, we randomly choose from three fonts, as seen in Figure E. For the color of the tags, we randomly choose from 4 colors: yellow, green, blue, and pink. Also, we randomize over 4 colors for the color of the pen: black, red, purple, and brown. We randomly choose these elements in advance. The dataset contains 100 categories and 1000 images. We use iPhoneX's camera, and the size of images is 3024 pixels by 3024 pixels. The code and dataset will be publicly available.

D. PAINT

In Sec. 4.2, we compare our method with PAINT [8]. For training for PAINT, we train the model for 3 epochs (2400 iterations) with batch size 16 using learning rate 1e-5 with 200 warm-up steps with a cosine annealing learning rate schedule and the AdamW optimizer (weight decay 0.1), following the paper.

E. Extended results on all datasets.

In tables B and C, we report the accuracy obtained on each of the 10 individual datasets for original and synthetic typographic attacks respectively.

Figure C. **Images sampled from our typographic attack COCO dataset.** The dataset consists of images from COCO with synthesized text.

Figure D. **Sample images from our real-world typographic attack dataset RTA-100.** The dataset contains 1000 images composed of 100 categories.

F. Visualization

To visualize the changes in word information, we generate images conditioned on text prompts using VQGAN+CLIP [3]. Fig. F presents samples of generated images: the first row shows images generated with original VQGAN+CLIP, capturing

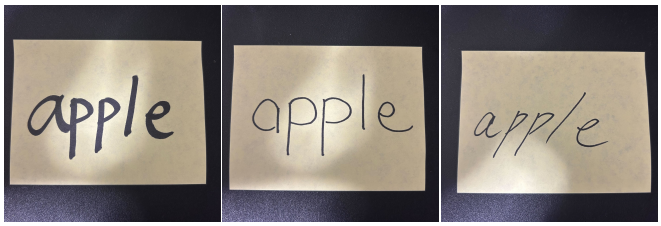

Figure E. **Sample images of fonts we used.** We use three fonts to write text: bold, normal, and italic.

Table B. **Classification results on all original datasets**

Method	ImageNet	Caltech	Pets	Cars	Flowers	Food	Aircraft	DTD	SUN	SAT	Avg.
CLIP	62.02	88.64	87.35	58.72	66.32	84.14	18.99	44.57	61.74	42.98	61.55
Materzynska+ [12]	54.38	80.53	75.01	40.33	51.86	55.01	13.23	36.28	51.06	37.32	49.50
PAINT [8]	61.82	88.48	85.23	55.30	**64.73**	80.51	17.73	**42.61**	**61.69**	38.20	59.63
Ours	**62.48**	**89.28**	**87.22**	**57.47**	63.82	**83.65**	**19.26**	40.64	61.41	**43.85**	**60.91**

Table C. **Classification results on all synthetic typographic attack datasets**

Method	ImageNet	Caltech	Pets	Cars	Flowers	Food	Aircraft	DTD	SUN	SAT	Avg.
CLIP	39.10	63.97	58.95	21.02	31.32	56.27	10.83	25.53	34.02	4.86	34.59
Materzynska+ [12]	44.91	74.73	63.61	15.79	34.95	43.41	8.28	33.03	39.52	16.22	37.44
PAINT [8]	**55.9**	**83.57**	**76.53**	**33.44**	**54.92**	**72.94**	14.46	**36.60**	**53.62**	**17.31**	**49.93**
Ours	49.83	79.54	72.88	28.64	44.12	67.79	**14.49**	31.6	43.50	9.65	44.20

Figure F. **Generated images conditioned on text prompts using VQGAN+CLIP.** Originally, CLIP often generates text of prompts as it is (top row) (e.g., "peas", "corn", "flower"). CLIP+Ours does not generate prompt texts in images, showing nonsense strings (bottom row).

the visual concepts of the prompt texts. In cases of "peas", "corn", and "flower", the images show the words of the prompts. The images generated with VQGAN+CLIP+Ours can also capture the visual concepts and do not show prompt text; instead, they show nonsense strings. The experiment demonstrates words with DP lose little original meanings, ruining the text information.

References

[1] Lukas Bossard, Matthieu Guillaumin, and Luc Gool. Food-101 – mining discriminative components with random forests. *ECCV*, 2014.

[2] Mircea Cimpoi, Subhransu Maji, Iasonas Kokkinos, Sammy Mohamed, and Andrea Vedaldi. Describing textures in the wild. *CVPR*, 2014.

[3] Katherine Crowson, Stella Biderman, Daniel Kornis, Dashiell Stander, Eric Hallahan, Louis Castricato, and Edward Raff. Vqgan-clip: Open domain image generation and editing with natural language guidance. *ECCV*, 2022.

[4] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. *CVPR*, 2009.

[5] Li Fei-Fei, Rob Fergus, and Pietro Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. *CVPR*, 2004.

[6] Agrim Gupta, Piotr Dollár, and Ross Girshick. Lvis: A dataset for large vocabulary instance segmentation. *CVPR*, 2019.

[7] Patrick Helber, Benjamin Bischke, Andreas Dengel, and Damian Borth. Eurosat: A novel dataset and deep learning benchmark for land use and land cover classification. *IEEE GRSS*, 2019.

[8] Gabriel Ilharco, Mitchell Wortsman, Samir Yitzhak Gadre, Shuran Song, Hannaneh Hajishirzi, Simon Kornblith, Ali Farhadi, and Ludwig Schmidt. Patching open-vocabulary models by interpolating weights. *NeurIPS*, 2022.

[9] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. *CVPR*, 2013.

[10] Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollár. Microsoft coco: Common objects in context. *ECCV*, 2014.

[11] Subhransu Maji, Esa Rahtu, Juho Kannala, Matthew Blaschko, and Andrea Vedaldi. Fine-grained visual classification of aircraft. *arXiv preprint arXiv: 1306.5151*, 2013.

[12] Joanna Materzyńska, Antonio Torralba, and David Bau. Disentangling visual and written concepts in CLIP. *CVPR*, 2022.

[13] Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. *ICVGIP*, 2008.

[14] Omkar M Parkhi, Andrea Vedaldi, Andrew Zisserman, and C. V. Jawahar. Cats and dogs. *CVPR*, 2012.

[15] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. *ICML*, 2021.

[16] Jianxiong Xiao, Krista A Ehinger, James Hays, Antonio Torralba, and Aude Oliva. Sun database: Exploring a large collection of scene categories. *IJCV*, 2016.