Improving Image Recognition by Retrieving from Web-Scale Image-Text Data

Supplementary Material

Ahmet Iscen Alireza Fathi Cordelia Schmid

Google Research

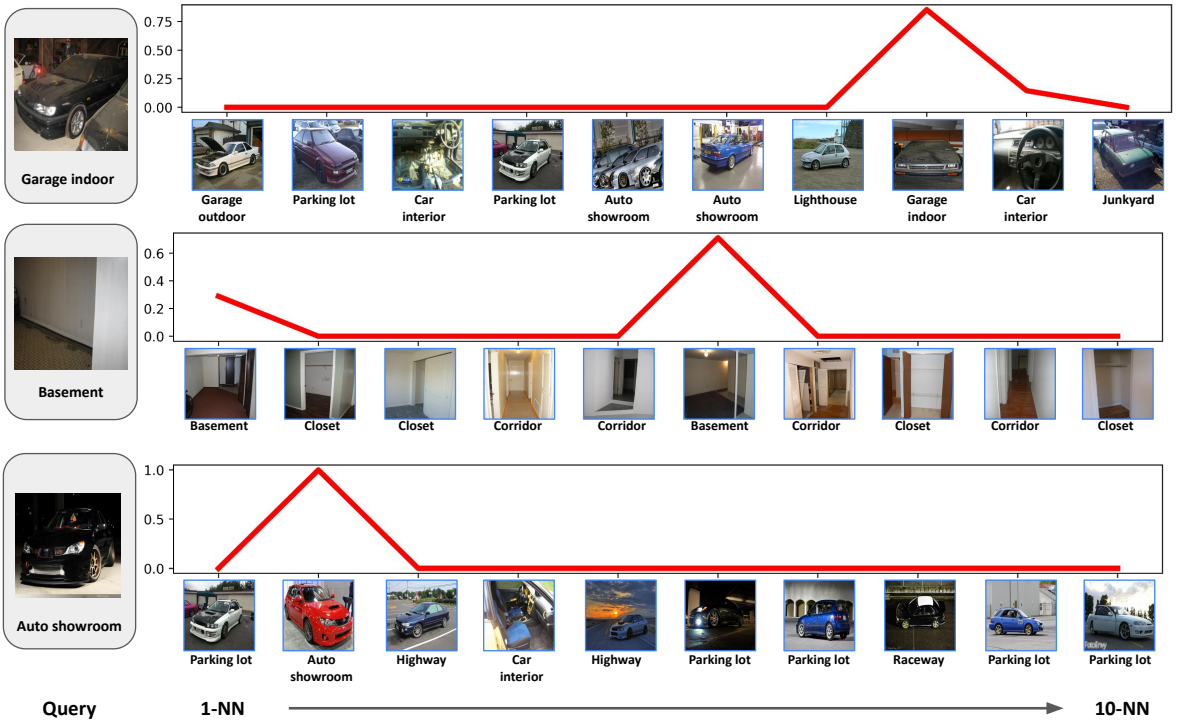

Figure 5. **Qualitative Examples.** We present the output of our method visually. We conduct this experiment by choosing the Places-LT dataset as the query and memory dataset. We display the query images from the test set on the left. Their k-NN images from the memory set are displayed on the right, and ordered from left to right. We display the attention weight assigned to each k-NN above the corresponding image.

A. Qualitative examples for Places-LT

Similar to Figure 4, we present some of the qualitative examples of the Places-LT dataset in Figure 5. We use Places-LT dataset as both the downstream task and the memory dataset for this experiment. We display the query images on the left, and the top-10 k-NN on the right.

We observe that our method differentiates between different environments containing *car* images. For example, for the *garage indoor* query, our method assigns a higher attention weight to the only other car image which is taken in a *garage indoor*, and filters out other images of cars taken in *garage outdoor*, *parking lot etc*.. We see a similar behavior for the *auto showroom* query, where the images of cars taken in *parking lots* are filtered out, and the only other image of a car in an *auto showroom* receives a high attention weight.

For the *basement* query, we see that two other *basement* images receive higher attention weights. In this case, attention weights also explain how the prediction is made. The *basement* image which is more visually similar to the query receives a higher attention weight. This demonstrates the potential of *explainability* of our method, from which we can derive how the decisions are made.

B. Qualitative comparison with Linear

We compare our method against the *Linear* baseline qualitatively on Table 2. We now present qualitative experiments for this comparison on Figure 6. For each query on the left, we display our correct prediction in green, and incorrect *Linear* prediction in red. We also display which category of classes, *i.e.* low-shot, mid-shot, many-shot, each of these queries belong to.

The first query is a challenging example of a *soda bottle*,

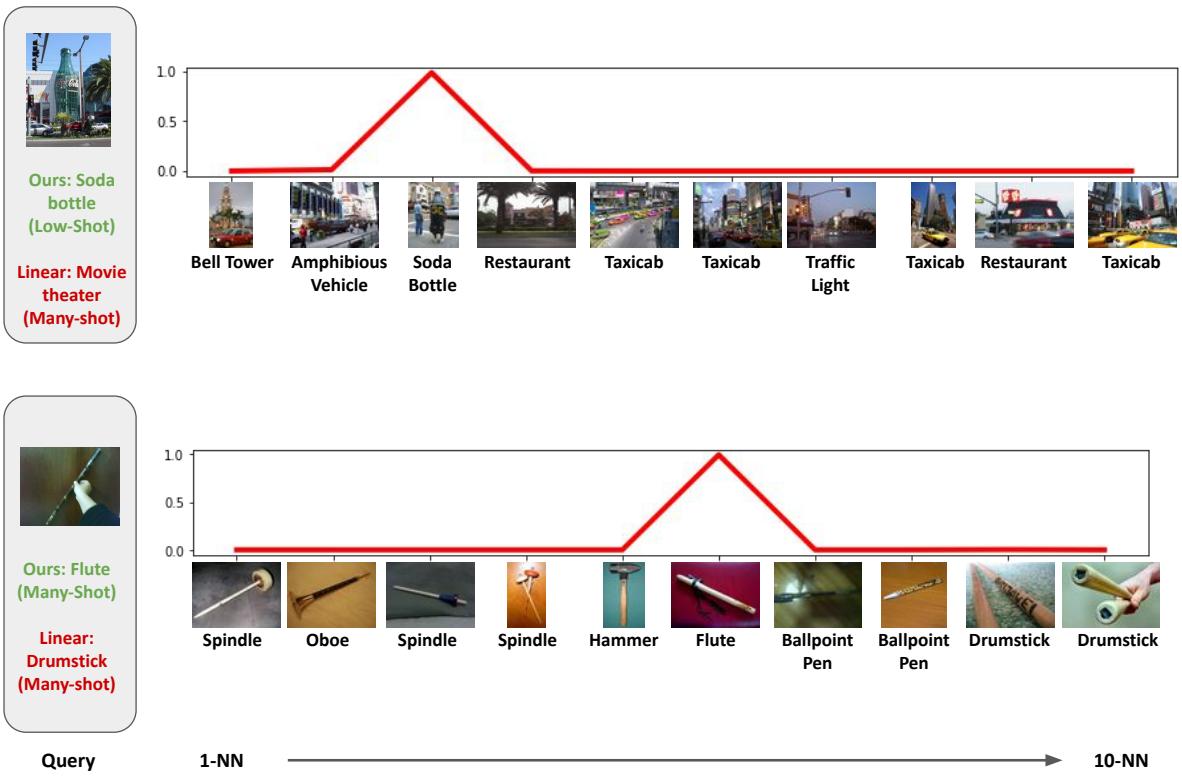

Figure 6. **Visual comparison on ImageNet-LT.** We present a visual comparison of our method and *Linear*. Below each query on the left, the correct prediction of our method is displayed in green, and the incorrect prediction of *Linear* is displayed in red. The k-NN images from the memory set are displayed on the right, and ordered from left to right, and their corresponding attention weights are displayed above them.

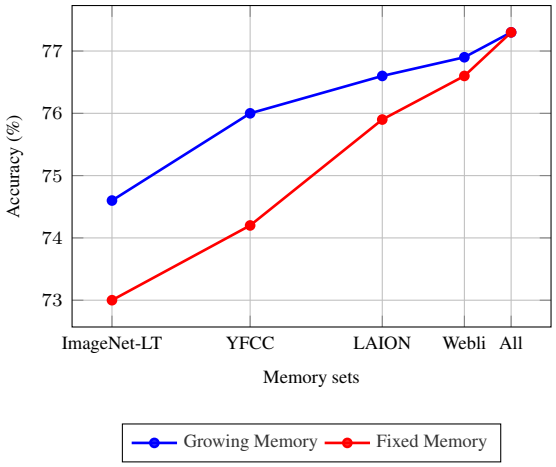

Figure 7. **Impact of growing memory on ImageNet-LT.** We train our method using the memory set denoted in the x-axis. For *growing memory*, additional data is added to the memory after the training, and the inference is done with the *All* memory. For *fixed memory*, training and the inference is done with the same memory.

which is much bigger than a typical *soda bottle*. Thus, the linear classifier incorrectly assigns a more frequently seen *movie theater*, which is a many-shot class, as its prediction, due to the building next to it. On the other hand, our method correctly predicts the *soda bottle* class, by retrieving another normal-than-usual soda bottle image in its k-NN list.

The second example, *flute*, belongs to a many-shot class. It is a visually challenging query, as it is similar to other objects such as *spindle* and *drumstick*. Our method predicts the correct class by retrieving another *flute* instance, whereas k-NN and *Linear* incorrectly predict *spindle* and *drumstick*, respectively.

C. Growing memory

We now present a different scenario, where the size of the memory dataset grows during the inference. More specifically, we train our method with either ImageNet-LT, YFCC, LAION or Webli as the memory set. After the training, we add more image-text pairs to our memory set and it becomes *All*. The memory attention module is not re-trained after adding additional data to the memory. Figure 7 shows that we can achieve higher accuracy without any extra training

cost, by just adding image-text pairs to the memory during the inference. The gains are more significant if our method is trained with a small memory set, *e.g*. ImageNet-LT, but the inference is done with much larger memory set, *i.e*. All.