Table 3. Evolution of example queries for three different initialization methods: K-LLM, K-Random, and K-Medoids. The table shows how the queries progress from the initial state, through a mid-training checkpoint, to the final post-training result.

Initialization	**Mid-Training**	**Post-Training**
Initialization Method: K-LLM		
reins	mustang	zebra stripes shining
wide mouth	feeding mouth	freight maul
driver	driver on the road	van on the road
Initialization Method: K-Random		
wood or gas fueled	large antelopes	horns blaring
photo-shop edits	flight stripes	grey feathers
travel distance	horns sound	body with horns
Initialization Method: K-Medoids		
lighting strike	zebra stripes	striped manes
ceiling beam	vintage sedan	station wagon body
knee bending motion	large animal moving	tractor trailer rig

A. Proof

In the following we provide the proof for Proposition 1.

Proof. First, we split the minimization between the dictionary parameters and V-IP network parameters:

$$\min_{\theta,\psi,\eta} J_{Q_\theta}(\psi,\eta) = \min_{\theta,\psi,\eta} \mathop{\mathbb{E}}_{X,S} \left[\ell_{Q_\theta,\psi,\eta}(X,S)\right] \tag{26}$$

$$= \min_{\theta} \min_{\psi,\eta} \mathop{\mathbb{E}}_{X,S} \left[\ell_{Q_\theta,\psi,\eta}(X,S)\right]. \tag{27}$$

Then we use that the cardinality of sampled query-answer histories S is uniformly distributed:

$$\min_{\theta} \min_{\psi,\eta} \mathop{\mathbb{E}}_{X,S} \left[\ell_{Q_\theta,\psi,\eta}(X,S)\right] = \min_{\theta} \min_{\psi,\eta} \frac{1}{K} \sum_{\tau=0}^{K-1} \mathop{\mathbb{E}}_{X,S:\,|S|=\tau} \left[\ell_{Q_\theta,\psi,\eta}(X,S)\right]. \tag{28}$$

Finally, we apply Proposition 1 from [5], which says that an optimal V-IP querier returns the IP queries and an optimal V-IP classifier equals the true posterior:

$$\min_{\theta} \min_{\psi,\eta} \frac{1}{K} \sum_{\tau=0}^{K-1} \mathop{\mathbb{E}}_{X,S:\,|S|=\tau} \left[\mathsf{D}_{\mathrm{KL}}\left(P\left(Y \mid X\right) \| P_\psi\left(Y \mid S, A_\eta\left(X,S\right)\right)\right)\right] \tag{29}$$

$$= \min_{\theta} \frac{1}{K} \sum_{\tau=1}^{K} \mathop{\mathbb{E}}_{X} \left[\mathsf{D}_{\mathrm{KL}}\left(P\left(Y \mid X\right) \| P\left(Y \mid \mathrm{Expl}^{\mathrm{IP}_\tau}_{Q_\theta}(X)\right)\right)\right] \tag{30}$$

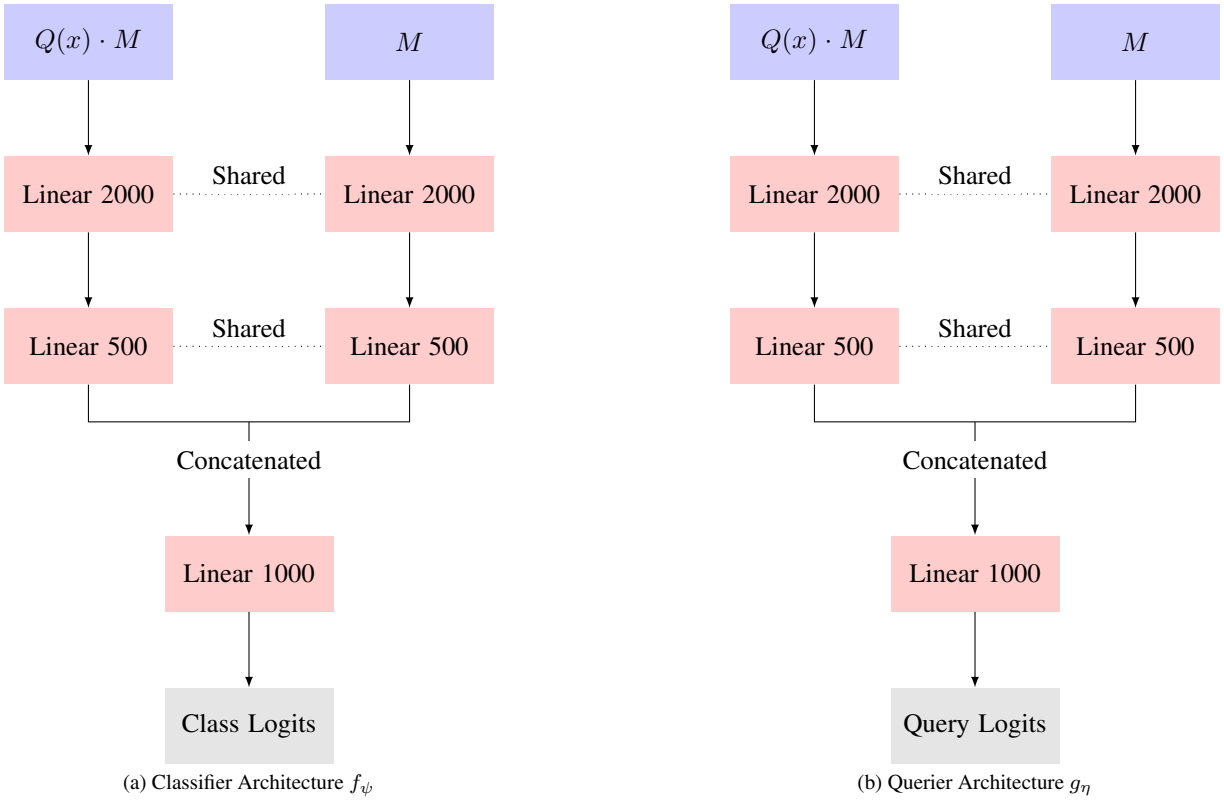
□

B. Classifier and Querier Network Architecture

The classifier and querier architecture is a two-layer, fully connected neural network. Figure 5 illustrates the architecture with a diagram. The size of the query dictionary only affects the final output dimension of the querier g_η, while the number of class labels only affects the final output dimension of the classifier f_ψ. We never share the weights between the classifier and the querier networks. We apply a softmax layer to the class and query logits to obtain probabilities for each class and query, respectively. During training, we employ a straight-through softmax [3] for the querier network.

Dataset	Budget τ	Full Method	w/o Query Answer Quantization	w/o Query Quantization
RIVAL-10	10	98.73%	99.49%	99.62%
CIFAR-10	10	95.12%	96.92%	97.42%
CIFAR-100	50	75.20%	82.45%	79.93%
ImageNet-100	50	83.99%	87.73%	84.92%
CUB-200	100	74.52%	81.65%	77.54%
Stanford-Cars	100	82.39%	87.20%	82.06%

Table 4. Ablation study comparing accuracy at fixed query budget of our full method (K-Learned) to variants without query answer quantization and query quantization. All runs use K-LLM to initialize the learnable dictionary. Quantization of query answers and queries improves interpretability while widening the gap to black-box methods.



(a) Classifier Architecture f_ψ (b) Querier Architecture g_η

Figure 5. Diagram of the neural network architecture for (a) classifier f_ψ and (b) querier g_η. "Shared" indicates that two linear layers share weights. "Concatenated" implies the output from previous layers is concatenated. Every arrow $\rightarrow$ before the concatenation and after the input layer applies a LayerNorm, followed by ReLU. In the forward pass of g_η, we convert the query logits into a one-hot encoding with a straight-through softmax.

C. Representing and Updating Query-Answer Histories

In V-IP, the input to the classifier f_ψ and querier g_η is a query-answer pair history S of variable length. Following the original V-IP work [5], we represent the history S for a sample x, as a binary mask M and $Q(x) \odot M$, where $\odot$ denotes the Hadamard product. If history S contains the i-th query-answer pair, then $M_i = 1$, otherwise $M_i = 0$.

Suppose $S^{(k)}$ denotes a history of k query-answer pairs for x and $M^{(k)}$ is the associated binary mask, then we can update

the history with a new query using the querier g_ψ:

$$M^{(k+1)} = M^{(k)} + g_\eta(S^{(k)}), \tag{31}$$

$$S^{(k+1)} = S^{(k)} + Q(x) \odot g_\eta(S^{(k)}). \tag{32}$$

In particular, g_η returns a one-hot encoding to select the next query. We use a straight-through softmax layer on the query logits to backpropagate gradients through g_η.

D. Training Details

For the baseline K-LLM dictionary, we follow the training schedule from [5] to train the V-IP querier and classifier networks with a fixed dictionary. For all datasets, we train the querier and classifier parameters jointly for 1500 epochs, starting with random sampling for query histories, followed by 1500 epochs of biased sampling, using the fixed K-LLM dictionary. We train our K-Learned dictionary with alternating optimization using $t = 4$ as the ratio of V-IP network to dictionary updates. For all datasets, we then train first for 800 epochs with random sampling for query histories, followed by 800 epochs using biased sampling. For the ablation study we train K-Learned with joint gradient updates in dictionary, querier, and classifier parameters, starting with 1500 random sampling epochs, followed by 1500 biased sampling epochs. Throughout all experiments we use Adam [16] as the optimizer with a learning rate of 1e-5 (other parameters use the default PyTorch values). We apply the straight-through estimator [3] in two places: (1) to backpropagate through the binarization of query answers in Eq. (25) and (2) to backpropagate through the nearest-neighbor operation in the query parameterization in Eq. (18). Hyperparameters were tuned on the AUC of validation accuracy as a function of the query budget, *i.e.*, the AUC of the curves in Fig. 3. During training we validate the AUC every 10 epochs and compare models at the best validation accuracy. In the ablation experiments we initialize our method with the K-LLM dictionary.

E. Query Universe Construction

In our experiments, $\mathcal{U}$ is formed as the union of all K-LLM dictionaries across tasks, supplemented by a generic query set, which is generated by prompting LLaMA-3B [13] with image captions from the COCO dataset [20], requesting 20 visual queries per caption. The prompt template for the COCO caption queries is shown below:

```
You receive as input an image caption.
You will output text snippets describing visual concepts that could
be present in the image.

I will give you two examples with the format I want you to use.

For example for the caption
"a group of zebras grazing in the grass"
you might output:

1. Stripes
2. Mammal
3. Tails
4. Four legs
5. Horizon line
6. Trees in the background
7. Grazing movement
8. Dust kicked up by movement
9. Wildlife in the background
10. Footprints in the grass
11. Blue sky

For example for the caption
"three airplanes sitting on top of an airport tarmac"
you might output:
```

1. jet engine
2. cloudy sky
3. wing
4. airport terminal

Each output element represents a"visual concept" that could be present in the image.
I will further give you examples of "good" and "bad" visual concepts.

Good visual concepts:
1. natural scenery
2. vegetable basket
3. hoofed animal
4. four-legged animal
5. brown coat
6. long legs
7. animal with horns
8. long neck
9. four-wheeled vehicle
10. flying object
11. stop sign
12. person riding vehicle
13. person riding horse
14. farm
15. barn
16. glass
17. metal
18. red color
19. blue color
20. green color
21. yellow color
22. car seat
23. urban scenery
24. flowing water
25. rainy weather
26. parking lot
27. wildlife
28. small animal
29. large animal
30. open road
31. city street
32. stairs
33. table with food
34. plate with food
35. traffic
36. long ears
37. textured surface
38. shiny surface
39. shiny object
40. shiny metal
41. city skyline
42. twigs and leaves

Bad visual concepts with explanations:
1. plane lights or navigation lights -> don't use "or"
2. office decor (walls, flooring, etc.) -> don't use "()"
3. different zebra patterns -> be concise and use zebra pattern

Please avoid "or" in your output as well as "()" in your output.
Keep the output concise and to the point and do not use more than
5 words per visual concept.

Only output the elements, no other text. Use the format
from the examples above.
Output maximum {} visual concepts per caption.

If a caption is inappropriate due to
violence or sexual content, output INAPPROPRIATE.