Hybrid Concept Bottleneck Models

Supplementary Material

Model	Dataset	number of classes	number of Images		
			Train	Dev	Test
Translator	MSCOCO	80	566,747	N/A	202,654
	ConceptNet	N/A	595,294	N/A	N/A
	ConceptGenerate	N/A	576,944	N/A	N/A
CBM	Food-101	101	50,500	20,200	30,300
	FGVC-Aircraft	100	3,334	3,333	3,333
	Flower-102	102	4,093	1,633	2,463
	CUB-200-2011	200	3,994	2,000	5,794
	UCF-101	101	7,639	1,898	3,783
	DTD	47	2,820	1,128	1,692
	HAM10000	7	8,010	1,000	1,005
	RESISC45	45	3,150	3,150	25,200
	CIFAR-10	10	45,000	5,000	10,000
	CIFAR-100	100	45,000	5,000	10,000
	ImageNet	1,000	1,281,167	50,000	N/A

Table 6. Detailed statistics of the datasets.

A. Resources

A.1. Dataset

Table 6 provides detailed statistics for all datasets. For concept translator pre-training, we use the entire ConceptNet [42] dataset, concepts generated by an LLM (ConceptGenerate), and the training split of MSCOCO [4]. In HybridCBM training, each dataset is annotated with a one-word description indicating its super class. We follow the train/dev/test splits provided by CoOp [56] for Food-101, Aircraft, Flower-102, UCF-101, and DTD. For CUB, we randomly sample 10 images per category as the development set. For CIFAR-10 and CIFAR-100, 10% of the training data is set aside for development. For HAM10000, we use an 80/10/10 split across classes, and for ImageNet, we evaluate only on the development set.

B. Implementation Details

B.1. Linear Probe

Following CLIP's implementation, we use encoded images (prior to projection into the vision-text embedding space) as input to the classifier, allowing us to evaluate how well the visual features alone can perform in classification tasks. We employ cuML's L-BFGS logistic regression, which is efficient for large-scale data, with a maximum of 1,000 iterations to ensure convergence.

To determine the optimal L2 regularization strength C, we perform a binary search on the validation set, beginning with a range of values $[1e^6, 1e^4, 1e^2, 1, 1e^{-2}, 1e^{-4}, 1e^{-6}]$, following the approach used in CoOp. Once the bounds of C are established, we iteratively refine the interval by halving it over 8 steps, ensuring a precise selection of the final hyperparameter value for improved model generalization.

Dataset	Method	Number of Shots					
		1	2	4	8	16	Full
CUB	Linear Probe	48.57	62.06	73.96	79.44	83.33	84.55
	Hybrid (submodular)	**53.81**	**65.30**	73.96	78.75	**83.75**	84.25
	Hybrid (random)	**51.66**	**64.22**	73.92	78.46	83.05	84.12
CIFAR-10	Linear Probe	73.61	86.36	92.37	95.19	96.00	98.05
	Hybrid (submodular)	**90.80**	**92.06**	**95.73**	**95.81**	**96.62**	97.93
	Hybrid (random)	**90.56**	**92.24**	**95.70**	**95.92**	**96.78**	98.04
CIFAR-100	Linear Probe	46.27	57.43	69.26	76.08	80.24	87.14
	Hybrid (submodular)	**47.98**	**57.82**	68.84	75.24	79.36	86.22
	Hybrid (random)	**47.96**	**57.71**	68.95	75.74	79.59	86.40
ImageNet	Linear Probe	41.16	54.21	64.76	71.17	75.26	84.37
	Hybrid (submodular)	38.33	51.00	62.11	68.59	73.65	83.67
	Hybrid (random)	37.61	49.34	61.86	68.64	73.57	83.62
Food-101	Linear Probe	61.94	77.14	83.97	87.98	89.98	93.15
	Hybrid (submodular)	**68.66**	**81.85**	**85.99**	87.50	89.34	92.62
	Hybrid (random)	**68.08**	**78.79**	**86.01**	87.37	89.41	92.60
DTD	Linear Probe	33.33	53.66	59.69	68.68	74.17	81.09
	Hybrid (submodular)	**45.27**	**62.00**	**64.18**	**70.86**	73.23	**81.21**
	Hybrid (random)	**44.98**	**61.23**	**64.36**	**71.16**	73.70	**81.26**
Flower-102	Linear Probe	83.52	94.07	97.60	98.78	99.39	99.47
	Hybrid (submodular)	**84.82**	92.94	97.04	97.89	98.86	99.23
	Hybrid (random)	**84.73**	92.98	96.51	98.05	98.86	99.19
UCF101	Linear Probe	61.51	76.63	78.43	84.64	88.16	89.82
	Hybrid (submodular)	**67.22**	**80.39**	**82.16**	**86.33**	**88.77**	**90.14**
	Hybrid (random)	**67.83**	**80.70**	**82.10**	**85.62**	88.16	**89.97**
Aircraft	Linear Probe	28.92	34.71	42.33	48.45	56.02	62.86
	Hybrid (submodular)	**30.21**	**36.03**	**44.58**	**50.92**	**58.69**	**64.92**
	Hybrid (random)	**30.87**	**36.36**	**43.74**	**50.20**	**57.49**	**65.05**
HAM10000	Linear Probe	34.33	42.39	45.57	64.58	64.58	81.49
	Hybrid (submodular)	**47.26**	**66.77**	**66.67**	**67.56**	**69.85**	**82.19**
	Hybrid (random)	**41.09**	41.39	**62.99**	60.60	60.80	**81.89**
RESISC45	Linear Probe	62.15	71.92	83.44	86.60	89.96	93.28
	Hybrid (submodular)	**70.63**	**78.53**	**84.65**	86.39	89.06	92.77
	Hybrid (random)	**70.99**	**78.59**	**84.72**	86.46	89.34	92.93

Table 7. Comprehensive results for Linear Probe and HybridCBM with a dynamic concept ratio of 0.5, evaluated on the test sets across 11 datasets.

Method	Number of Shots	Ratio of Dynamic Concepts					
		0.2	0.4	0.5	0.6	0.8	1
Hybrid (submodular)	1	39.0	40.9	44.0	42.2	40.6	33.8
	2	42.5	41.4	40.3	43.9	38.8	35.0
	4	43.1	41.8	42.3	42.7	39.9	38.5
	8	44.7	43.6	43.7	42.2	41.1	38.5
	16	44.1	42.3	43.6	44.6	40.4	37.7
	Full	42.2	44.6	**46.2**	43.3	40.1	39.0

Table 8. Ablation precision@t for varying the ratio of dynamic concepts under different number of shots.

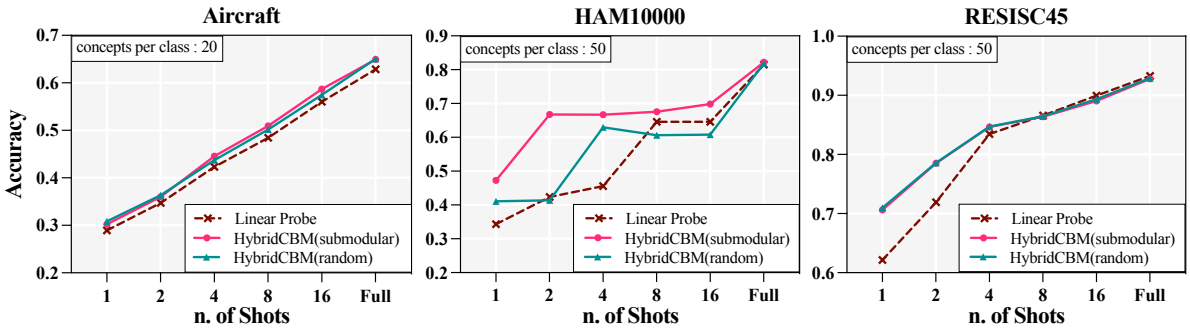

Figure 6. Comparison of test accuracy between HybridCBM with various static concept selections and Linear Probe across 3 datasets. The x-axis denotes the number of labeled images.

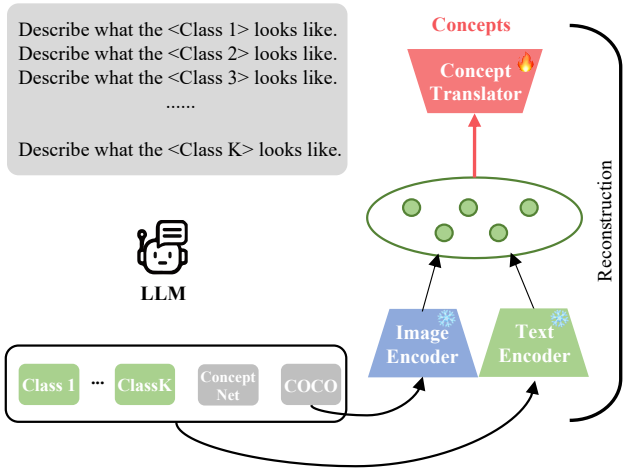

Figure 7. **Concept Translator Pre-training.** The LLM is prompted to generate concepts, which are then combined with COCO captions to train the Concept Translator.

B.2. Concept Translator

Following the methodologies of ClipCAP [24] and DeCAP [19], we leverage CLIP's shared feature space by training a concept translator from scratch. This translator bridges the gap by converting embeddings from CLIP's image or text space into human-interpretable concepts. Suppose an embedding-concept pair (e, c), where $c = \{word_1, word_2, \ldots, word_n\}$, the translator $\mathcal{T}$ learns to reconstruct c conditioned on the preceding words and the l_2-normalized image or text embedding e. The training objective is:

$$\mathcal{L}_{\text{Recons}} = -\frac{1}{n}\sum_{i=1}^{n} \log P_{\mathcal{T}}(word_i \mid word_{<i}, e) \qquad (10)$$

As shown in Figure 7, to enable diverse and accurate translation of vectors to concepts, we collect a large concepts corpora consisting of all generated concepts by ChatGPT, a train-subset of MSCOCO [4] captions and the entire set of concepts from ConceptNet [42]. We download a pre-built list of all edges (assertions) from ConceptNet 5.7 and retain only those entries containing the "surfaceText" field within the JSON structure. This field provides a human-readable representation of the assertion as concepts. For example: "[[Cat]] is an [[animal]]."Then, we utilize the image encoder of CLIP to encode the image-caption pairs of MSCOCO as embedding-caption pairs and the text encoder to encode the large concepts corpora as embedding-concept pairs. The translator maps the concept embedding e to the textual description c through $c = \mathcal{T}(e)$. This mapping enhances interpretability by providing semantic meanings to the learned concepts. As noted, we train the concept translator from scratch using this large corpus and a cross-entropy loss. We utilize GPT-2 [31], configured with a 12-layer Transformer [44]. The hidden state size is set to 768, matching the dimensions of CLIP's shared feature space. We evaluate our concept translator on the MSCOCO test split.

B.3. Prompt

Figure 8 presents the prompts used to query GPT-4 for computing the *Semantic Concept Validation* metric. To assess the alignment between concepts and their corresponding classes, we prompt the LLM to act as a concept classifier. For each concept, the LLM is asked to determine whether the concept belongs to a specified class, responding with "yes" if it does and "no" otherwise. This process allows us to measure the semantic validity of the learned concepts, as determined by GPT-4's responses.

Temperature is a parameter that controls the "creativity" or randomness of the text generated. A higher temperature (e.g., 0.7) results in more diverse and creative output, while a lower temperature (e.g., 0.2) makes the output more deterministic and focused. In practice, we set the temperature to 0 to make the model completely deterministic, always choosing the most likely token.

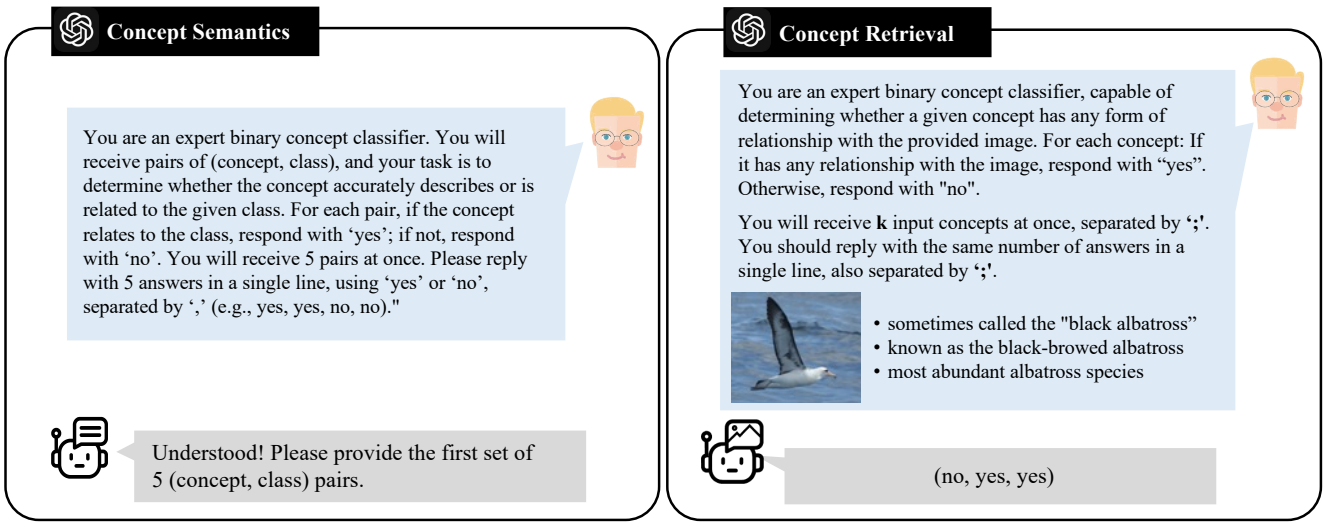

Figure 8. Detailed GPT-4 concept evaluation Prompts. This part describes the prompts for computing *Semantic Concept Validation* metric.

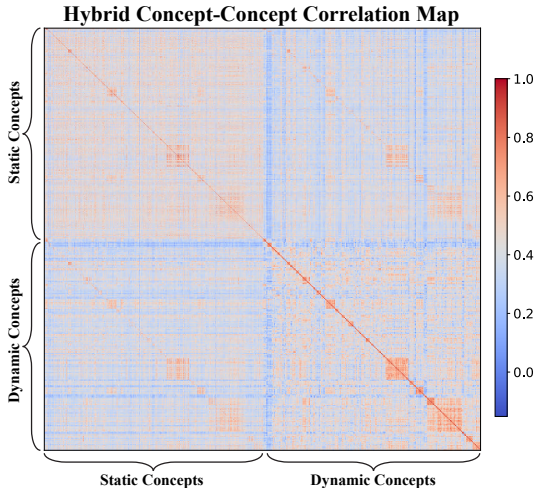

Figure 9. Hybrid Concept-Concept Correlation Map for the CUB Dataset. The x-axis and y-axis represent selected static and dynamic concepts, respectively, showcasing the correlation structure across these concept types.

Dataset	α	β	λ_{align}
CUB-200-2011	2	0.1	0.01
FGVC-Aircraft	0.5	0.1	0.01
CIFAR-10	0.5	0.5	0.01
CIFAR-100	0.5	0.5	0.01
Other	1	0.1	0.01

Table 9. Hyperparameters used in the experiments. "Other" refers to the remaining datasets not explicitly listed.

B.4. Full Results

The full numerical results are shown in Table 7. The test accuracy are provided.

B.5. Case Interpretability

To evaluate the case interpretability of dynamic concepts, we utilize a Vision-Language Model (GPT-4o), which effectively measures both the semantic quality and interpretability of our model. As shown in Table 8, we evaluated interpretability performance by varying both the ratio of dynamic concepts and the number of shots on the CUB dataset. The results demonstrate that setting the dynamic concept ratio to 0.5—meaning the number of static and dynamic concepts are equal—leads to better model case interpretability. Notably, even in few-shot scenarios, our model achieves a comparable precision@t (greater than 0.4), highlighting the effectiveness of dynamic concepts in enhancing case-level understanding under limited data conditions. However, increasing the dynamic concept ratio beyond 0.8 leads to a decrease in precision@t. This is probably due to the limitations of the concept translator arising from distribution bias when there are few static concepts. To address these limitations, future work could explore advanced concept translation techniques or incorporate adaptive weighting between static and dynamic concepts to balance their influence.

B.6. Hybrid Concept-Concept Correlation

We also present the Concept-Concept Correlation Map of the hybrid concept bank in Figure 9, which is composed of static and dynamic concepts. The map reveals stronger intra-group correlations (top-left and bottom-right) and weaker inter-group correlations (off-diagonal), highlighting distinct clustering and complementary relationships between static and dynamic concepts.

C. Other Details

Resources. Our model is implemented using PyTorch Lightning. We use CLIP from OpenAI's official repository. All experiments are run on a combination of 8 NVIDIA RTX 3090 GPUs and 4 NVIDIA RTX 4090 GPUs.

Running Time. With CLIP's weights frozen, we extract image features once and reuse them across all experiments, significantly reducing training time. Since we only optimize a set of concept vectors and a linear layer, our model is highly efficient. The additional computational overhead of our method is minimal, with the most intensive operation being the Sinkhorn divergence calculation, which typically scales quadratically with the size of the concept banks due to iterative optimization.

Hyperparameters. The proposed training scheme utilizes five hyperparameters: α, β, λ_{dis}, λ_{ortho}, and λ_{align}. Throughout all experiments, we fix λ_{dis} to 1 and λ_{ortho} to 0.1. The values of the remaining three hyperparameters, customized for each dataset, are summarized in Table 9.

Optimization. We employ two Adam optimizers [16]: one for optimizing dynamic concepts with an initial learning rate of 1×10^{-3} and another for training the sparse linear layer with an initial learning rate of 5×10^{-5}.

Sinkhorn divergence. The Sinkhorn divergence is computed as:

$$\mathfrak{S}_{div,\epsilon}(\mu,\nu) = \mathfrak{S}_{\epsilon}(\mu,\nu) - \frac{1}{2}\left(\mathfrak{S}_{\epsilon}(\mu,\mu) + \mathfrak{S}_{\epsilon}(\nu,\nu)\right) \quad (11)$$

where $\mathfrak{S}_{\epsilon}$ represents the Sinkhorn distance between μ and ν with regularization parameter ϵ. We use an efficient GPU implementation provided by https://www.kernel-operations.io/geomloss/ library.