Verbalized Representation Learning for Interpretable Few-Shot Generalization

Supplementary Material

A. Implementation Details

In this section, we outline the detailed prompt template used to generate verbalized features, and the hyperparameters used throughout the experiment. Since the proposed verbalized representation learning (VRL) only involves inference using Vision Language Models, we are able to significantly improve the inference speed and the GPU memory usage by leveraging existing optimization techniques. Specifically, we utilize Sglang [53], which introduces optimizations such as RadixAttention for KV cache reuse to accelerate inference. In our experiments, we use LLaVA-OneVision as the VLM, since it is able to interleave multiple images in the prompt. For GPU usage, the 7B model requires 2 A6000 GPUs, each with 48GB of RAM, while the 72B model demands 8 A6000 GPUs to host the model.

A.1. Prompt Templates in VRL

We report the prompt template used to generate verbalized features capturing inter-class difference (y_{diff}) and intra-class commonality (y_{comm}) in Table 6. Notably, since the generated descriptions often include multiple features, we utilize the same VLM again to parse the descriptions into a question and the corresponding answers, as shown in Table 7. This approach enables us to disentangle the various features captured by the VLMs, making them easier to map to scalar vectors. Consequently, to extract the representation of a given image using the learned verbalized features, we prompt the VLM to determine whether the described features are present in the image, as discussed in Sec. 3.1. The prompt used for this stage is detailed in Table 8. For inter-class difference features (y_{diff}), we assign a value of 0 if the model identifies the image as more similar to class 1 for the given attribute, and 1 if it is more similar to class 2. For intra-class commonality features (y_{comm}), we assign a value of 1 if the model responds with 'Yes' and 0 if it responds with 'No'.

```
{
   "role": "user",
   "content": [
      {
         "type": "image_url",
         "image_url": {
            "url": "data:image/jpeg;base64,
                {{image1}}"
         },
         "modalities": "multi-images"
      },
      {
         "type": "image_url",
         "image_url": {
            "url": "data:image/jpeg;base64,
                {{image2}}"
         },
         "modalities": "multi-images"
      },
      {
         "type": "text",
         "text": q_diff/q_comm
      }
   ]
}

q_diff = "Identify the most distinctive
     feature that can be used to distinguish
     the species between image 1 and image 2."
q_comm = "List the key features that not only
     shared by the species in both images but
     also make this species distinct from
     others. Focus on unique or specific
     characteristics, such as detailed patterns
      in the arrangement, textures, color
     variations, or specific forms of growth on
      surfaces. Provide each feature as a
     distinct bullet point, capturing the
     essence of what makes this species
     visually identifiable."
```

Table 6. Prompt template for generating verbalized features. Note that q_{diff} is the text query used to generate inter-class difference feature y_{diff} and q_{comm} is for intra-class commonality y_{comm}.

A.2. Time Complexity

Given a classification task with C classes and N-shot examples per class, we are able to construct $C_2^C \times C_2^N$ and $C \times C_2^N$ pairs for inter-class and intra-class images, respectively. For example, with 5 classes and 10 images per class, we can sample 450 and 100 distinct pairs for inter-class and intra-class cases. In addition, even with the same image pairs, perform sampling during VLM's generation can also produce diverse verbalized features. Theoretically, to learn a set of verbalized features, the time complexity is $O(b \cdot C_2^C \cdot C_2^N)$ for y_{diff} and $O(b \cdot C \cdot C_2^N)$ for y_{comm}, where b represents the number of samples generated by the VLM for the same image pair. While it may sound intimidating, empirically, we find that setting b to 1 and perform verbalized representation learning on $C \times N$ pairs for both inter and intra cases are sufficient to learn a diverse robust features set. Specifically, for a task involving 5 classes with 10 images per class, this requires only 50 inferences, which can be completed in under 30 seconds. After obtaining the verbalized features, each training image must be mapped into numeric representations based on the learned features y_{diff} and y_{comm}. Given that there are $C \times N$ images, and

```
system_prompt (y_diff) = """
I have a series of descriptions that I would like to convert into classification questions. For each
     description, respond in JSON format, which includes a question and provides specific labels for
     Class 1 and Class 2 based on the key distinguishing feature mentioned in the description.
\nExample description: The most distinctive feature that can be used to distinguish class 1 and
    class 2 is the type of fungus present. class 1 has a bright yellow, fuzzy fungus with a round
    shape, while class 2 has bright yellow, delicate flower-like structures growing from a dark gray
     tree branch.
\nExample response: {\"question\": \"What type of fungus is present?\", \"class_1\": \"bright yellow
    , fuzzy fungus with a round shape\", \"class_2\": \"bright yellow, delicate flower-like
    structures growing from a dark gray tree branch\"}
"""

system_prompt (y_comm) = """
I have a series of descriptions that I would like to convert into a list of structured sentences,
    where each item describes one specific feature of the species. For each description, response in
     a list format.
\nExample description: The berry in both images exhibits several distinctive characteristics that
    set it apart from other berry species:\n\n- **Flower Structure**: The flowers are small, with
    five petals each, and they form in clusters. The petals are delicate and appear to be a soft
    pink or white color.\n- **Leaf Arrangement**: The leaves are arranged in an opposite or
    alternate pattern, with each leaf having a distinct shape that is often described as oval with a
     pointed tip.\n- **Leaf Texture**: The leaves have a velvety texture, which is unique to this
    species.\n- **Stem and Branches**: The stems and branches have small thorns or are spiny, which
    can be a defense mechanism against herbivores.\n- **Foliage Color**: The foliage is a vibrant
    green, indicating a healthy, thriving plant.\n- **Berries**: The berries are small, round, and
    appear to be a dark red or purple color, typical of many berry species.\n- **Growth Environment
    **: Both images show the plant growing in a rocky, perhaps alpine environment, which suggests it
     has adapted to grow in challenging conditions.\n- **Unique Shape**: The leaves and flowers have
     a unique shape, with the leaves having a slightly wavy edge and the flowers having a bell-
    shaped form.
\nExample response: [\"Its flowers are small, with five petals each, and they form in clusters. The
    petals are delicate and appear to be a soft pink or white color.\",\"The leaves are arranged in
    an opposite or alternate pattern, with each leaf having a distinct shape that is often described
     as oval with a pointed tip.\",\"The leaves have a velvety texture, which is unique to this
    species.\",\"The stems and branches have small thorns or are spiny, which can be a defense
    mechanism against herbivores.\",\"The foliage is a vibrant green, indicating a healthy, thriving
     plant.\",\"The berries are small, round, and appear to be a dark red or purple color, typical
    of many berry species.\",\"The plant growing in a rocky, perhaps alpine environment, which
    suggests it has adapted to grow in challenging conditions.\",\"The leaves and flowers have a
    unique shape, with the leaves having a slightly wavy edge and the flowers having a bell-shaped
    form.\"]
"""

user_prompt = f"Now, convert this description: {y_diff/y_comm}" + " Please follow the same JSON
    format for the response. Response:"
```

Table 7. Given the verbalized feature (y_{diff} and y_{comm}), we use the VLM to convert the description into a question and the corresponding answer for each class.

each image is evaluated against $C \times N$ descriptions, the computational complexity of this stage is $O(C^2 \cdot N^2)$. Empirically, for a task with 5 classes and 10 images, our approach requires less than 30 seconds on a single A6000 to extract verbalized features. In contrast, LLM-Mutate [12] based on text-based LLM sampling could take 22 hours as the generated features often lack visual grounding, resulting in slower convergence on discriminative features.

It is worth noting that to accelerate the feature mapping process, we can replace generative VLMs like LLaVA with encoder models like CLIP to perform similarity-based feature mapping, as discussed in Sec. 3.1. Since we can perform similarity computation in a two-dimensional batch-wise operation, where one dimension encapsulates all the images while the other contains all the verbalized features. As a result, the time complexity is reduced to $O(1)$, which finishes in seconds, albeit with a slight trade-off in performance, as demonstrated in Table 5.

A.3. Hyperparameters

In this subsection, we outline the specific parameters used to construct the visual classifiers. For implementation, we

```
user_prompt (y_diff) = f"Given the following
    image, classify it based on the provided
    criteria:
\nCriteria (Question): {question}
\nClass 1: {class_1_ans}
\nClass 2: {class_2_ans}
\nPlease response with \"Class 1\" or \"Class
    2\"

user_prompt (y_comm) = f"Examine the given
    image and determine if it matches the
    features described by the following
    criteria: {question}. Answer only with YES
     or NO."

{
   "role": "user",
   "content": [
      {
         "type": "image_url",
         "image_url": {
            "url": f"data:image/jpeg;base64,{
                image}"
         },
      },
      {
         "type": "text",
         "text": user_prompt,
      },
   ],
}
```

Table 8. Prompt template used to map verbalized feature (y_{diff}, y_{comm}) to numeric representations (F_{diff}, F_{comm}).

utilized the scikit-learn [37] package. We use the default parameters for all classifiers. Since the primary results are based on logistic regression and multi-layer perceptron (MLP) classifiers, we provide the detailed parameters for these here and refer readers to the official scikit-learn documentation for details on other classifiers. For logistic regression, regularization was applied using the ‘l2’ norm via the penalty parameter. The solver was ‘lbfgs’, suitable for multiclass problems, and the regularization strength was controlled by C, set to 1.0. Optimization stopping criteria were determined by ‘tol’ with the default value of 0.0001.

For MLP classifier, the network has a single hidden layer with 100 neurons and uses the ReLU activation function. Optimization is handled by the Adam solver with a learning rate of 0.001 and an L2 regularization term controlled by alpha=0.0001. The model trains for a maximum of 200 iterations with a batch size set automatically, which is theminimum of 200 and the number of training samples. Early stopping is disabled and the tolerance for optimization convergence is 0.0001.

F_{diff}	F_{comm}	**CLIP**	**DINO**	**Avg.**
✓				65.26
	✓			58.32
		✓		63.26
			✓	64.06
✓	✓			67.92
		✓	✓	72.86
✓	✓	✓		76.52
✓	✓		✓	76.86
✓	✓	✓	✓	79.92

Table 9. Accuracy (%) when incorporating feature vectors learned from different methods. F_{diff} and F_{comm} denote the difference and commonality feature vectors learned from VRL. CLIP and DINO refer to the image features encoded by CLIP and DINO visual encoder, respectively. All results are reported using the ensemble of the best-performing classifier combinations. Specifically, F_{diff}, F_{comm}, and DINO are using logistic regression while CLIP features are classified by MLP classifier.

B. Additional Analysis

B.1. Feature Fusion with existing visual encoders

Intuitively, verbalized representation learning can be viewed as a fine-tuning process where we develop features specifically tailored to our few-shot data, but without requiring gradient update steps. To validate this perspective, we investigate whether the learned verbalized features can enhance the performance of pretrained visual encoders.

Table 9 presents the results of combining difference (F_{diff}) and commonality (F_{comm}) features with features extracted by pre-trained visual encoders (CLIP and DINO). These features are incorporated via an ensemble approach, where each feature is used to train a separate classifier, and the final prediction is determined by averaging the prediction logits from all classifiers.

From the table, we observe that when features predict individually, the performance hovers around 60%, with F_{diff} yielding the highest accuracy among the standalone features. When features are combined, significant performance improvements are achieved. Specifically, adding both verbalized features (F_{diff} and F_{comm}) to CLIP features leads to a notable accuracy increase of 13.26%, while a similar 12.8% improvement is observed when combining these features with DINO.

Finally, combining all four features–F_{diff}, F_{comm}, CLIP, and DINO–results in a peak performance of 79.92%. This validates the effectiveness of integrating verbalized features with pre-trained visual embeddings, demonstrating that verbalized representation learning provides complementary, task-specific refinements that significantly enhance model performance in few-shot learning scenarios.

B.2. Ablation Study

We present the complete ablation study of our method in Table 10, analyzing the performance across several dimen-

Size	F.T.	F.M.	LR	RF	SVM	kNN	NB	DT	GB	MLP
7b	y_{diff}	LLaVA	52.73	49.33	47.27	45.80	50.20	40.07	47.33	51.73
7b	y_{comm}	LLaVA	53.27	51.27	51.33	45.80	53.53	40.07	43.33	50.87
7b	both	LLaVA	62.06	56.00	51.40	51.20	52.06	37.80	39.60	55.80
7b	y_{diff}	CLIP	57.07	55.13	48.47	46.33	47.53	43.20	45.60	52.73
7b	y_{comm}	CLIP	46.53	50.47	45.53	43.40	19.33	38.80	42.00	56.67
7b	both	CLIP	48.07	49.47	45.53	43.40	21.73	38.40	41.93	58.87
72b	y_{diff}	LLaVA	65.27	64.93	57.07	56.07	53.53	45.13	53.60	62.53
72b	y_{comm}	LLaVA	58.33	58.07	53.60	51.47	52.20	38.73	48.47	58.33
72b	both	LLaVA	67.40	64.87	58.93	54.47	55.33	41.73	44.67	66.00
72b	y_{diff}	CLIP	61.07	57.13	54.07	47.00	53.60	42.60	46.13	57.20
72b	y_{comm}	CLIP	45.87	50.13	47.80	44.33	19.33	40.47	35.87	53.73
72b	both	CLIP	53.67	52.80	50.40	44.73	19.33	42.93	40.93	60.60

Table 10. Comparison of classification accuracy (%) across different ablated methods for fine-grained classification on iNaturalist. Note that F.T. indicates the type of the verbalized features and F.M. refers to the model used to perform feature mapping. For different classifiers, LR denotes Logistic Regression, RF for Random forest, SVM for Support Vector Machine, kNN for k nearest neighbor, NB for Naive Bayes, DT for decision tree, GB for gradient boosting and MLP for multi-layer perceptron classifier.

sions. Specifically, we evaluate our model using two different sizes (7B and 72B), the impact of distinct verbalized features (y_{diff} and y_{comm}), and the effect of using different feature mapping models (LLaVA or CLIP). Additionally, we examine the effectiveness of various classifiers, including logistic regression (LR), random forest (RF), support vector machine (SVM), k-nearest neighbor (kNN), naive Bayes (NB), decision tree (DT), gradient boosting (GB), and multi-layer perceptron (MLP). We observe that larger models (72B) consistently outperform smaller models (7B) across most classifiers and settings, showcasing the benefit of increased model capacity for capturing verbalized features. We also find that the inter-class difference features (y_{diff}) are generally more effective than commonality feature. However, we discover a consistent trend where the combined features (via concatenation) can yield the best overall performance (the 'both' rows). For different feature mapping models, LLaVA outperforms CLIP in most scenarios, showcasing the advantage of using generative VLMs to determine the presence of a certain feature. In terms of classifier, we observe that logistic regression, random forest and MLP classifiers perform the best. On the other hand, we notice that decision tree is prone to overfitting on the training set since we only have few-shot samples, while Naive Bayes also struggle to perform well since the resulting representations are high-dimensional.

B.3. Mini-ImageNet

In the main paper, to evaluate how well the proposed method learn when the objects are not well-presented in the pre-training datasets of the VLMs, we conduct experiments on the iNaturalist and Kiki-Kouba datasets for fine-grained and novel object recognition. To complement these experiments and provide insight into its performance on a more general few-shot benchmark, we further evaluate our method on the mini-ImageNet dataset under the 5-way 1-shot setting across 1,000 testing episodes. We report the results in Table 11.

Ours	F. Liu [30]	M. Liu [31]	Chen [9]
94.27 ± 0.05	98.24	81.14 ± 0.15	72.31 ± 0.40

Table 11. Top-1 Accuracy on mini-ImageNet under the 5-way 1-shot setting. Baseline results are sourced from the original paper.

LLaVA	10	50	200	LLaVA	10	50	200
w/ SFT	43.9	65.2	80.2	w/ VRL	62.1	80.5	86.2

Table 12. Scaling LLaVA with SFT vs. VRL from 10 to 200 (full dataset) training images per class.

From the table one can see that our method is comparable to the recent baselines that adapt VLMs or LLMs for few-shot image classification. Notably, these baselines typically require data from training episodes to perform meta-training or instruction fine-tuning, while our method directly adapts during test time without further training.

B.4. Scalability Analysis

Scaling training data. We evaluate the proposed VRL framework on progressively larger training sets and summarize the results in Table 12. Our method consistently outperforms standard supervised fine-tuning—even when using the full dataset (200 images per class). We attribute this advantage to the fact that, unlike the random weight initialization in SFT, verbalized queries provide a strong prior that guides the model toward learning task-relevant

	Acc.	**Rel.**	**Rel. (%)**
Lichen	71.6	18.0	90
Wrasse	72.0	20.0	100
Wilde Rye	74.0	20.0	100
Manzanita	56.0	17.0	85
Bulrush	66.0	18.0	90
Average	67.9	18.6	93

Table 13. Human Study of the learned verbalized features on iNaturalist. Acc. denotes testing classification accuracy, while Rel. represents human evaluation scores (maximum 20).

features. Importantly, we find that exhaustive pairwise enumeration is not required for these gains: sampling only 100 image pairs (out of 19,900 possible) still delivers substantial improvements. For feature extraction, we adopt the CLIP-based VRL variant (see L269–274), which enables efficient feature mapping without compromising performance.

Scaling object classes. We evaluate scalability on a large iNaturalist subset with 200 families, each with 5 species (1000 species) and 10 images per species. For family-level classification, which is more heterogeneous, CLIP achieves 73%, while our $F_{diff} + F_{comm}$ features reach 83%. Combining them further improves performance to **90%**. Similar trends are observed in the general recognition dataset miniImageNet (Table 12, Appendix), where VRL achieves **94.2%**. It suggests that general VLM features can handle coarse, heterogeneous distinctions well, and since VRL builds on top of them, it naturally inherits this capability while providing additional gains by extracting finer, task-relevant features. Notably, in this experiment, we sample only 1 image pair per class combination, yet still observe substantial improvements.

B.5. Human Study on Verbalized Features

To verify the quality of the learned verbalized features, we sample 20 features per super class on iNaturalist, resulting in 100 features for human evaluation. We ask the testers to evaluate whether the generated features are faithful to the image and relevant to the target objects. For example, if a feature accurately describes the image but pertains only to the background (e.g., "*the sky is blue*"), it receives a score of 0. We report the results in Table 13. From the results one can see that around 93% of the features were faithful to the image and useful for classification, with a 0.84 correlation between this rate and final accuracy. In addition, we observe that the learned classifier tend to assign lower weights for those irrelevant or hallucinated features, thereby reducing their impact on final predictions.