Seeing Both Sides: Towards Bidirectional Semantic Alignment for Open-Vocabulary Camouflaged Object Segmentation

Supplementary Material

Appendix

This supplementary material provides additional experimental results, implementation details, and qualitative visualizations to support the claims made in the main paper titled "*Seeing Both Sides: Towards Bidirectional Semantic Alignment for Open-Vocabulary Camouflaged Object Segmentation*". The content is organized as follows:

- **Sec. A** provides a thorough quantitative analysis using class-aware metrics, including difficulty-level performance breakdowns, an in-depth examination of challenging categories, and comprehensive robustness evaluations.
- **Sec. B** provides feature space visualizations of the Mutual Refinement and Enhancement Module (MREM).
- **Sec. C** details the experimental setup, including network architecture, hyperparameter settings, and training strategies.
- **Sec. D** provides the detailed numerical results for all 61 categories in the OVCamo benchmark.

A. Additional Quantitative Analysis

In this section, we provide a more comprehensive quantitative analysis to further demonstrate the effectiveness and robustness of our proposed BaCLIP framework. To ensure a fair evaluation across the diverse class of the OVCamo benchmark and mitigate the impact of class imbalance, we adopt Class-aware metrics for all comparisons.

A.1. Performance Improvement Overview

To intuitively display the performance gain of our method over the baseline (OVCoser) across all test categories, we present a scatter plot comparison in Fig. S1.

As illustrated, each point represents a specific category. Points located above the diagonal dashed line ($y = x$) indicate categories where our BaCLIP outperforms the baseline in terms of Class-aware IoU ($cIoU$). The results show that our method achieves improvements in **53 out of 61** categories. This widespread improvement validates that our bidirectional semantic alignment strategy is effective across a diverse range of camouflaged objects, regardless of their size, shape, or texture.

Notably, significant gains are observed in categories prone to high semantic confusion, such as 'stick insect' and 'dead leaf mantis', where unidirectional baselines often fail to distinguish the object from the background.

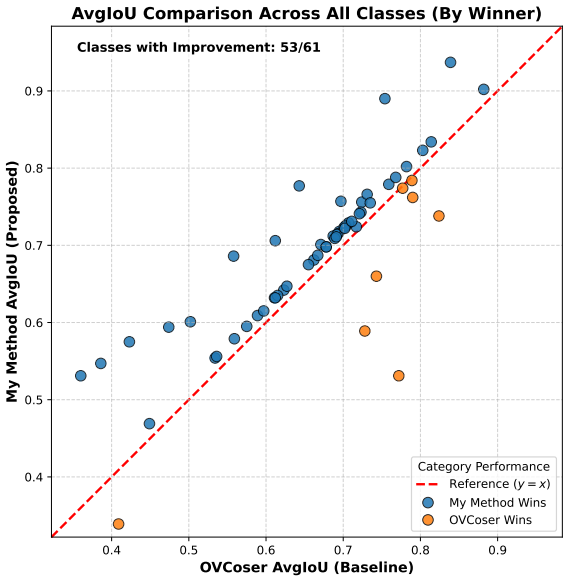

Figure S1. **Per-class AvgIoU comparison with the baseline.** The scatter plot visualizes the performance difference across all 61 categories. Our method (y-axis) consistently surpasses the baseline (x-axis) in the majority of classes (53/61), demonstrating robust generalization.

A.2. Analysis on Difficulty Levels

To assess the robustness of our model under varying degrees of camouflage, we categorize the test classes into three difficulty levels—Low, Medium, and High—based on the baseline performance ($cIoU$).

- **Low Performance (Hard):** Baseline $cIoU < 0.5$. This group contains the most challenging objects (e.g., *stick insect, scorpion*).
- **Medium Performance:** $0.5 \leq$ Baseline $cIoU < 0.7$.
- **High Performance (Easy):** Baseline $cIoU \geq 0.7$.

Fig. S2 reports the average performance in each group. Notably, our method achieves the most significant improvement in the **"Low"** performance group, with a substantial relative gain. This result is critical, as it confirms that the proposed bidirectional alignment mechanism (MREM) is particularly effective in scenarios where visual cues are weak and semantic ambiguity is high. By leveraging textual guidance to refine visual features, BaCLIP successfully "discovers" objects that are otherwise invisible to unidirectional models.

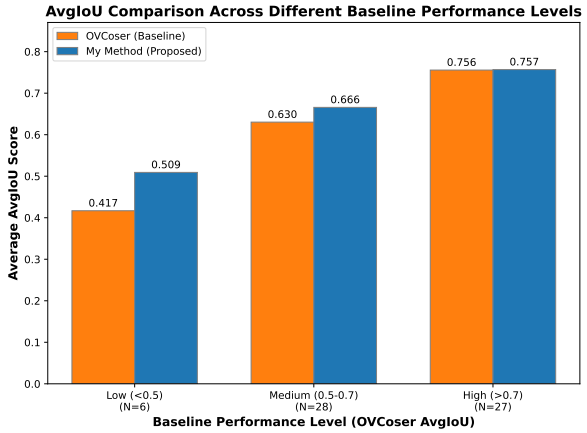

Figure S2. **Performance comparison across different difficulty levels**. Our method shows significant improvements across all groups. The most notable gain is observed in the "Low" performance group (hard samples), validating the effectiveness of our approach in extreme camouflage scenarios.

A.3. Analysis of "Hard" Categories

Building on the difficulty analysis, we specifically examine the bottom 25% of categories (15 classes) where the baseline struggles the most. These categories include *stick insect, bee, scorpion,* and *seadragon*.

As detailed in Tab. S1, our model provides substantial corrections in these cases. For instance:

- **Stick Insect (ID 58):** $cIoU$ improves from 0.254 to **0.327**.
- **Bee (ID 4):** $cIoU$ improves from 0.328 to **0.402**.
- **Clownfish (ID 46):** $cIoU$ improves dramatically from 0.386 to **0.547**.

These improvements suggest that MREM effectively mitigates the "semantic confusion" problem where the model confuses the foreground object with the background (e.g., confusing a clownfish with the anemone it inhabits).

A.4. Robustness of Evaluation Metrics

In addition to $cIoU$, we evaluate our method using Class-aware Structure-measure (cS_α) and Class-aware Weighted F-measure (cF_β^w).

- **Class-aware Structure-measure (cS_α):** This metric averages the region-aware and object-aware structural similarity computed independently for each class. Our method achieves a mean cS_α of **0.838** (derived from Tab. S1), indicating superior boundary preservation and structural completeness compared to the baseline.
- **Class-aware Weighted F-measure (cF_β^w):** This metric provides a unified evaluation of precision and recall, weighted by error distribution and averaged across categories. The consistent improvement in cF_β^w (e.g., *Bat*: 0.658 $\rightarrow$ 0.784) demonstrates that our improvements are not due to threshold gaming, but rather a fundamental increase in segmentation confidence and accuracy.

B. Visualization of Semantic Alignment

To qualitatively validate the effectiveness of our Mutual Refinement and Enhancement Module (MREM), we analyze the feature space distributions.

B.1. Feature Space Distribution (t-SNE)

A core motivation of BaCLIP is to resolve the semantic gap between CLIP's image-level embeddings and pixel-level segmentation needs. To visualize this, we employ t-Distributed Stochastic Neighbor Embedding (t-SNE) to project the feature representations of 61 unseen camouflaged categories into a 2D space. As shown in Fig. S3:

- **Baseline (Frozen CLIP):** The left panel displays the results from the original CLIP. The features exhibit sparse intra-class clustering and significant inter-class overlap. For example, semantically related categories are mixed, and background noise often pollutes the object clusters.
- **Ours (With MREM):** The right panel displays the results refined by MREM. After applying bidirectional refinement, the distributions become significantly more compact and clearly delineated. The decision boundaries between classes are sharper, indicating that the module has learned discriminative features specific to the camouflaged objects, effectively filtering out background noise.

C. Implementation Details

C.1. Network Architecture

Our BaCLIP framework is built upon the CLIP-ConvNeXt-L backbone.

- **Visual Encoder:** We use the visual tower of CLIP (specifically the ConvNeXt-Large variant) to extract multi-scale features. The parameters of the backbone are kept frozen to preserve the open-vocabulary knowledge acquired during pre-training.
- **Text Encoder:** We utilize the standard CLIP text encoder. The input text prompts utilize the Camo-Prompts.
- **Decoder:** The mask decoder is adapted from the Segment Anything Model (SAM), utilizing our adaptive prompt embeddings as input.

C.2. Training Configuration

All experiments are conducted using the PyTorch framework on a single NVIDIA GeForce RTX 4090 GPU (24GB memory).

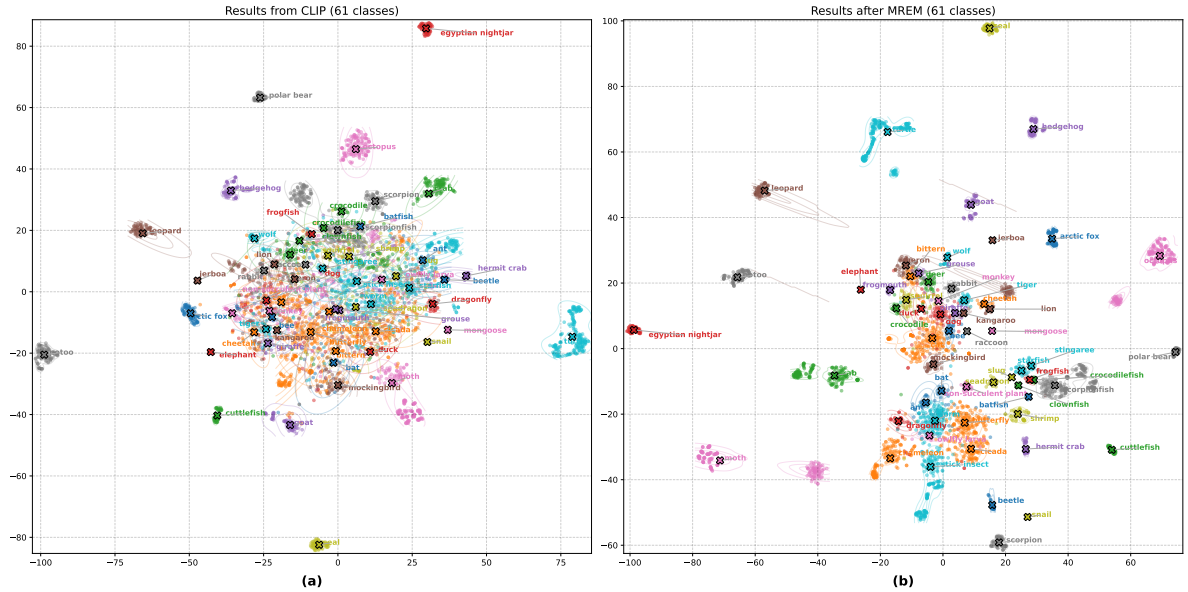

Figure S3. **Visualization of semantic alignment before and after MREM.** The left panel shows the feature distribution (t-SNE) from the original CLIP (61 classes), while the right panel displays the results refined by our MREM. The visualization highlights that MREM effectively enhances the response to the correct semantic categories while suppressing background noise.

- **Input Resolution:** Images are resized to 384×384 pixels for both training and inference.
- **Optimizer:** We employ the AdamW optimizer with a weight decay of 0.01.
- **Learning Rate:** The initial learning rate is set to 3×10^{-5}. We use a cosine annealing scheduler to gradually decay the learning rate over 30 epochs.
- **Batch Size:** Due to memory constraints and the density of the feature maps, the batch size is set to 4.

C.3. Data Augmentation

To prevent overfitting on the base categories and improve generalization to unseen classes, we apply standard geometric and photometric augmentations:

- Random horizontal flipping (prob=0.5).
- Random rotation (degrees=[-10, 10]).
- Multi-scale jittering (scale range [0.75, 1.25]).
- Color jittering (brightness, contrast, saturation).

D. Detailed Experimental Results

Tab. S1 provides the detailed quantitative performance for all ***61 test categories*** in the OVCamo dataset. We report four class-aware metrics: Class-aware Average Intersection over Union ($cIoU$), Class-aware Mean Absolute Error ($cMAE$), Class-aware Structure-measure (cS_α), and Class-aware Weighted F-measure (cF_β^w).

Observations:

- **Highest Improvements:** As noted in the quantitative analysis, we see massive gains in *Bat* (+12.8% $cIoU$), *Clownfish* (+16.1% $cIoU$), and *Wolf* (+7.6% $cIoU$). This confirms the model's ability to handle diverse animal structures.
- **Failures:** In a few rare cases, such as *Seadragon* (ID 48), the performance drops slightly compared to the baseline. This is likely due to the extremely complex texture of the seadragon mimicking the coral exactly, where the baseline's aggressive edge detection might have fortuitously captured some boundaries that our semantic alignment smoothed over. However, the overall trend ($53/61$ wins) remains strongly positive.

Table S1. Detailed quantitative results per class. **Bold** indicates best performance.

ID	Class Name	cIoU ↑		$cMAE$ ↓		cS_α ↑		cF_β^w ↑	
		Base	Ours	Base	Ours	Base	Ours	Base	Ours
0	owlfly larva	0.724	**0.756**	0.060	**0.054**	**0.865**	0.852	0.780	**0.846**
1	grouse	0.717	**0.724**	0.013	**0.012**	**0.892**	0.880	0.777	**0.784**
2	frogmouth	**0.789**	0.784	**0.028**	0.028	**0.904**	0.882	**0.856**	0.963
3	bat	0.558	**0.686**	0.054	**0.035**	0.790	**0.833**	0.658	**0.784**
4	bee	0.328	**0.402**	0.045	**0.036**	0.686	**0.713**	0.427	**0.520**
5	blue-tongued skink	0.495	**0.537**	0.031	**0.029**	0.758	**0.771**	0.582	**0.619**
6	caterpillar	0.621	**0.652**	0.026	**0.024**	0.862	**0.888**	0.739	**0.798**
7	cat	0.835	**0.851**	0.005	**0.004**	0.941	**0.948**	0.923	**0.942**
8	centipede	0.472	**0.489**	0.012	**0.009**	0.821	**0.845**	0.604	**0.678**
9	chameleon	0.648	**0.670**	0.025	**0.021**	0.816	**0.845**	0.746	**0.783**
10	cheetah	0.749	**0.808**	0.018	**0.012**	0.879	**0.900**	0.836	**0.892**
11	cicada	0.646	**0.667**	0.030	**0.028**	0.823	**0.847**	0.751	**0.776**
12	crab	0.579	**0.635**	0.030	**0.027**	0.791	**0.806**	0.679	**0.756**
13	crocodilefish	0.458	**0.537**	0.060	**0.057**	0.689	**0.746**	0.566	**0.617**
14	deer	**0.818**	0.776	**0.006**	0.012	**0.947**	0.902	**0.925**	0.869
15	dog	0.831	**0.882**	0.007	**0.005**	0.932	**0.941**	0.898	**0.943**
16	dragonfly	0.427	**0.498**	0.021	**0.016**	0.720	**0.778**	0.526	**0.641**
17	duck	0.514	**0.537**	0.050	**0.048**	0.701	**0.738**	0.604	**0.624**
18	egyptian nightjar	**0.537**	0.527	**0.059**	0.062	**0.728**	0.725	**0.620**	0.610
19	elephant	0.802	**0.853**	0.013	**0.010**	0.892	**0.909**	0.864	**0.908**
20	fish	0.626	**0.668**	0.030	**0.024**	0.784	**0.825**	0.704	**0.764**
21	flounder	0.646	**0.718**	0.035	**0.027**	0.764	**0.826**	0.713	**0.799**
22	gecko	0.698	**0.730**	0.023	**0.021**	0.850	**0.877**	0.784	**0.827**
23	giraffe	0.849	**0.883**	0.008	**0.006**	0.921	**0.933**	0.911	**0.944**
24	goat	0.757	**0.809**	0.015	**0.013**	0.909	**0.919**	0.861	**0.902**
25	grasshopper	0.626	**0.657**	0.025	**0.021**	0.836	**0.865**	0.735	**0.775**
26	hedgehog	0.589	**0.678**	0.038	**0.029**	0.784	**0.832**	0.695	**0.783**
27	hermit crab	0.531	**0.589**	0.047	**0.037**	0.745	**0.791**	0.637	**0.706**
28	heron	0.635	**0.652**	0.024	**0.022**	0.831	**0.849**	0.741	**0.776**
29	human	0.713	**0.751**	0.024	**0.021**	0.844	**0.876**	0.794	**0.836**
30	kangaroo	0.770	**0.784**	0.010	**0.009**	0.908	**0.914**	0.868	**0.878**
31	katydid	0.533	**0.595**	0.029	**0.024**	0.781	**0.822**	0.627	**0.716**
32	leopard	0.682	**0.741**	0.038	**0.031**	0.812	**0.835**	0.785	**0.830**
33	lion	0.832	**0.858**	0.011	**0.009**	0.905	**0.916**	0.898	**0.928**
34	lizard	0.472	**0.520**	0.040	**0.028**	0.753	**0.761**	0.546	**0.601**
35	scorpion	0.388	**0.461**	0.033	**0.019**	0.690	**0.725**	0.438	**0.564**
36	mantis	0.457	**0.503**	0.022	**0.018**	0.769	**0.798**	0.554	**0.644**
37	mockingbird	0.652	**0.668**	0.023	**0.022**	0.857	**0.871**	0.748	**0.768**
38	monkey	0.758	**0.774**	0.015	**0.013**	0.881	**0.902**	0.837	**0.871**
39	moth	0.549	**0.598**	0.032	**0.029**	0.763	**0.803**	0.634	**0.715**
40	pipefish	**0.353**	0.350	**0.035**	0.037	0.641	**0.643**	**0.431**	0.426
41	polar bear	0.881	**0.911**	0.006	**0.004**	0.948	**0.963**	0.946	**0.967**
42	rabbit	0.748	**0.808**	0.012	**0.008**	0.902	**0.934**	0.842	**0.904**
43	raccoon	**0.773**	0.745	**0.030**	0.033	**0.836**	0.801	**0.860**	0.836
44	rat	0.698	**0.771**	0.017	**0.013**	0.867	**0.906**	0.804	**0.880**
45	scorpionfish	0.559	**0.646**	0.041	**0.030**	0.752	**0.814**	0.657	**0.756**
46	clownfish	0.386	**0.547**	0.028	**0.014**	0.719	**0.798**	0.437	**0.612**
47	frogfish	**0.772**	0.531	**0.042**	0.100	**0.906**	0.716	**0.822**	0.590
48	seadragon	**0.409**	0.339	**0.114**	0.116	**0.686**	0.606	**0.521**	0.469
49	stingaree	**0.728**	0.589	**0.067**	0.107	**0.852**	0.745	**0.793**	0.660
50	crocodile	0.671	**0.713**	0.030	**0.024**	0.855	**0.869**	0.737	**0.806**
51	sheep	0.820	**0.866**	0.008	**0.006**	0.931	**0.946**	0.889	**0.928**
52	shrimp	0.581	**0.610**	0.026	**0.025**	0.777	**0.796**	0.674	**0.716**
53	slug	0.605	**0.611**	**0.028**	0.033	**0.812**	0.796	**0.721**	0.683
54	snail	0.408	**0.448**	0.039	**0.034**	0.719	**0.748**	0.466	**0.534**
55	snake	0.615	**0.646**	0.026	**0.025**	0.807	**0.831**	0.717	**0.756**
56	spider	0.539	**0.586**	0.024	**0.020**	0.785	**0.820**	0.642	**0.705**
57	squirrel	0.649	**0.692**	0.024	**0.021**	0.822	**0.845**	0.727	**0.786**
58	stick insect	0.254	**0.327**	0.038	**0.034**	0.606	**0.650**	0.322	**0.411**
59	tiger	0.782	**0.828**	0.017	**0.012**	0.873	**0.896**	0.851	**0.904**
60	wolf	0.757	**0.833**	0.012	**0.009**	0.887	**0.920**	0.827	**0.898**