FlowComposer: Composable Flows for Compositional Zero-Shot Learning

Supplementary Material

We provide additional details and experimental results in this supplementary material, which is organized as follows:

7. More Experimental Details

7.1. Dataset Statistics

For each dataset, we adhere to the data splitting scheme from [41, 43]. Detailed data statistics are reported in Tab. 6. MIT-States [17] is a large-scale attribute–object benchmark containing diverse everyday objects paired with a wide range of state attributes (*e.g.*, *sliced*, *moldy*, *wet*), resulting in a dense and visually heterogeneous composition space. UT-Zappos [52] focuses on fine-grained footwear recognition, where attributes primarily describe material or surface appearance (*e.g.*, *leather*, *satin*, *suede*) and objects correspond to specific shoe categories. C-GQA [39] extends the GQA dataset with compositional attribute–object annotations, covering a much broader variety of scenes, colors, textures, and object types, and thus presents a more diverse and challenging open-world composition landscape.

Table 6. Statistics of three datasets in our experiments. The number of elements in each set is reported.

			Training		Validation			Test		
Dataset	$\mathcal{S}$	$\mathcal{O}$	$\mathcal{C}^{se}$	$\mathcal{X}$	$\mathcal{C}^{se}$	$\mathcal{C}^{us}$	$\mathcal{X}$	$\mathcal{C}^{se}$	$\mathcal{C}^{us}$	$\mathcal{X}$
MIT-States [17]	115	245	1262	30k	300	300	10k	400	400	13k
UT-Zappos [52]	16	12	83	23k	15	15	3k	18	18	3k
C-GQA [39]	413	674	5592	27k	1252	1040	7k	888	923	5k

7.2. Additional Implementation Details

Consistent with prior studies [15, 41], we adopt the official OpenAI checkpoints of CLIP with a ViT-L/14 [45] configuration as our visual–textual backbone. For data preprocessing and training configuration, we follow the same setup as Troika [15] and CSP [41]: images are processed with the same resizing and cropping pipeline, and Adam is used as the optimizer for all datasets. For *FlowComposer*, we utilize AdamW [35] as the optimizer for both primitive flows and the composer.

Flow Matching Networks. As introduced in the main paper Sec. 5.1, We employ a lightweight architecture proposed in MAR [27] as flow matching network, implemented as a deep residual MLP with timestep conditioning. Concretely, the input feature $\boldsymbol{x} \in \mathbb{R}^D$ (CLIP feature of dimension D) is fed into a stack of 24 ResBlocks with adaptive LayerNorm (adaLN) modulation, where timestep embeddings from a small MLP (*TimestepEmbedder*) control per-channel shift, scale, and gating, and each block uses SiLU activations.

Composer Networks. We implement the *Composer* as a lightweight residual MLP that takes the attribute and object velocities $\boldsymbol{v}_a, \boldsymbol{v}_o \in \mathbb{R}^D$ as input, and processes them with LayerNorm, GELU activations, and a small stack of residual MLP blocks before projecting back to $\mathbb{R}^D$. We report the details of the model parameters in Tab. 7.

Table 7. Model size and computational cost of our networks.

Network	# Params (M)↓	GFLOPs↓
FM	74.06	0.07
Composer	25.97	0.03

8. Remained Feature Entanglement Analysis

8.1. Leakage Analysis

To further support our argument about *Remained Feature Entanglement*, we quantitatively probe the amount of cross-branch "leakage" in the multi-path baseline Troika [15]. Our hypothesis is that, under perfect disentanglement, the attribute branch should encode only attribute-related information and be essentially uninformative for object prediction (and vice versa). In that case, using attribute features to classify objects (or object features to classify attributes) should yield accuracies close to random chance. To test this, we extract the attribute, object, and composition features from Troika [15] on the test sets and use each branch to predict both attribute and object labels with the same CLIP-style cosine classifier aligned with Troika's inference. We then measure the resulting attribute and object accuracies for all three branches on all three datasets, and report the results in Tab. 8.

In Tab. 8 we report *class-balanced accuracy* (per-class average). The first row (Random) is a chance baseline with randomly generated predictions. The second row (*Attr Branch*) uses the *attribute* image features to classify both attributes and objects via the same CLIP-style cosine classifier; the third row (*Obj Branch*) does the symmetric test with *object* features; the fourth row (*Comp Branch*) uses the *composition* features. Clear leakage emerges: using attribute features to predict *objects* is far above chance (MIT: **3.87** vs. 0.47, UT: **14.9** vs. 8.14, C-GQA: **2.36** vs. 0.05).

Conversely, using object features to predict *attributes* also surpasses chance (MIT: **6.3** vs. 1.13, UT: **7.69** vs. 5.74, C-GQA: **2.12** vs. 0.25). Moreover, composition features are strongly predictive for both primitives (*e.g.*, object accuracy: MIT **51.3**, UT **58.9**, C-GQA **39.8**), indicating substantial cross-branch carry-over. These findings are inconsistent with perfect disentanglement and quantitatively support our claim of *remained feature entanglement*.

Table 8. Comparison between random prediction and predictions from *Attribute*, *Object* and *Composition* branches on class-balanced accuracy (*Acc*) for attributes and objects.

	MIT-States		UT-Zappos		C-GQA	
Acc(%)	Attr	Obj	Attr	Obj	Attr	Obj
Random	1.13	0.47	5.74	8.14	0.25	0.05
Attr Branch	26.3	3.87	22.3	14.9	3.64	2.36
Obj Branch	6.3	45.2	7.69	63.2	2.12	17.3
Comp Branch	21.3	51.3	21.6	58.9	9.72	39.8

8.2. Disentanglement Analysis

We further refine our analysis by examining the role of the visual disentangler itself. In our initial cross-branch probe, we observe that on certain datasets the attribute and object branches can even become less predictive when the disentangler suppresses information too aggressively, suggesting that disentanglement may sometimes remove useful cues together with entangled ones. To better understand whether the disentangler remains beneficial overall, we conduct an additional controlled study: using the same backbone and training protocol, in Tab. 9, we compare (i) **w/ Disentangler**, the standard multi-path baseline [15], against (ii) **w/o Disentangler**, a variant in which we remove the disentangler.

Across all datasets, removing the disentangler leads to clear drops in HM and AUC, indicating that feature disentanglement, although imperfect and not strictly factor-pure, still provides meaningful structural benefits for compositional generalization.

In summary, the cross-branch "leakage" accuracies remain far above random under both settings, reinforcing our central observation that *perfect* attribute–object separation is difficult to achieve in practice. This aligns with the motivation of our main paper: rather than pursuing an idealized, fully factorized representation, we embrace the remaining entanglement and turn it into a resource. Leakage-Guided Augmentation is therefore designed not to eliminate leakage, but to harness the surviving cross-factor signals as additional supervision, reflecting the philosophy that residual correlations, when properly guided, can be leveraged rather than suppressed.

Table 9. Comparison between Troika and Troika without disentangler in MIT-States [17], UT-Zappos [52] and C-GQA [39] on HM and AUC under the closed-world setting.

	MIT-States		UT-Zappos		C-GQA	
Disentangler	HM	AUC	HM	AUC	HM	AUC
w	39.2	22.1	55.4	41.8	29.7	12.5
w/o	36.8	20.4	51.4	37.5	28.0	11.2

9. Additional Analysis

9.1. Additional Component Analysis

We provide additional component analysis to quantify the contribution of each module: *Primitive Flows (Flows)*, *Composer*, and *Leakage-Guided Augmentation (LG-Aug)* under the ***open-world*** setting. As shown in Tab. 10, we report results on MIT-States [17], UT-Zappos [52], and C-GQA [39], comparing against the Troika [15] baseline. Starting from the baseline, we incrementally add our components and make four observations. **First**, introducing *Primitive Flows* (Row (1)) delivers a consistent boost, especially on seen accuracy, indicating that explicit transport toward textual endpoints helps beyond token-level composition. **Second**, adding *Leakage-Guided Augmentation* on top of Flows (Row (2)) further improves robustness on unseen pairs, yielding higher HM and AUC by leveraging residual cross-branch cues. **Third**, alternatively replacing LG-Aug with the *Composer* (Flows+Composer, Row (3)) also stabilizes recognition by providing an explicit combination rule in the embedding space, improving HM and AUC over Flows alone. **Fourth**, combining all components (**Ours**) achieves the best performance across all datasets and metrics, demonstrating that explicit velocity composition and leakage-guided supervision are complementary to flow learning in the open-world regime.

9.2. Computational Costs

Tab. 11 reports the inference-time comparison between two baselines (CSP [41] and Troika [15]) and our *FlowComposer*. We report the average latency (ms) per image over the entire test set. Our method increases latency only slightly (*e.g.*, from 17.6 to 19.2 ms per image on MIT-States [17], 11.0 to 12.7 ms per image on UT-Zappos [52], and 69.8 to 74.4 ms per image on C-GQA [39] for Troika [15]). This modest cost stems from our one-step transport scheme and lightweight Composer, indicating that the performance gains come with negligible computation.

9.3. Composition Stepsize Analysis

We introduce the hyperparameter stepsize h of the one-step composition in Sec. 4.2. As shown in the main paper Tab. 3, we evaluate $h \in \{0.1, 0.5, 1.0\}$ on MIT-States [17] and UT-Zappos [52] under the closed-world setting. The optimal h

Table 10. Ablations. The results regarding the different components in our *FlowComposer* on MIT-States [17], UT-Zappos [52] and C-GQA [39] under the open-world setting.

	Flows (§ 4.1)	*Composer* (§ 4.2)	*LG-Aug* (§ 4.3)	MIT-States [17] Seen	Unseen	HM	AUC	UT-Zappos [52] Seen	Unseen	HM	AUC	C-GQA [39] Seen	Unseen	HM	AUC
Baseline	✗	✗	✗	50.3	17.5	19.0	6.8	65.8	61.0	46.6	32.7	40.8	8.6	11.7	2.9
(1)	✓	✗	✗	50.1	18.0	19.6	7.0	68.5	60.8	49.3	33.8	41.2	9.0	12.0	3.2
(2)	✓	✗	✓	49.7	18.9	20.2	7.3	69.1	**61.5**	50.1	34.7	42.2	10.0	12.3	3.3
(3)	✓	✓	✗	49.6	18.8	20.0	7.2	69.8	61.1	50.7	34.9	41.9	**10.3**	12.2	3.2
Ours	✓	✓	✓	**50.4**	**19.0**	**20.3**	**7.5**	**70.1**	61.2	**51.0**	**35.5**	**43.5**	10.2	**12.6**	**3.5**

Table 11. Inference time (ms/image) comparison between two baselines and ours.

Method	MIT-States	UT-Zappos	C-GQA
CSP	6.6	10.3	20.2
+ Ours	9.3	12.5	24.6
Troika	17.6	11.0	69.8
+ Ours	19.2	12.7	74.4

differs across datasets: UT-Zappos [52] favors a larger step (h=1.0), whereas MIT-States [17] prefers smaller steps. We attribute this to dataset-specific composition geometry: UT-Zappos contains fewer, more separated compositions, where a longer step moves features closer to their text targets; in contrast, the denser composition spaces of MIT-States make large steps prone to overshooting and error accumulation. Accordingly, we adopt dataset-specific choices of h in all experiments.

Here, we provide an additional ablation study on C-GQA [39]. As shown in Tab. 12, the trend on C-GQA [39] mirrors the behavior observed in MIT-States [17] and UT-Zappos [52]. A smaller composition stepsize (h=0.1) yields the best performance across all four metrics on both the validation and test sets. Larger stepsizes (h=0.5 and h=1.0) consistently reduce HM and AUC, indicating that overly long updates tend to overshoot the target direction in C-GQA's dense composition space. The close alignment between validation and test results further confirms that the stepsize sensitivity is stable and not due to overfitting or evaluation noise. This provides additional support that different datasets exhibit distinct composition geometries, and that adopting dataset-specific stepsizes leads to more reliable one-step transport.

Table 12. Results regarding stepsize h (§ 4.2) on the validation and test set under the closed-world setting for C-GQA [39].

	Validation				Test			
h	S	U	H	A	S	U	H	A
0.1	**44.6**	**40.6**	**32.9**	**15.5**	**44.8**	**40.7**	**34.0**	**15.9**
0.5	44.6	39.9	32.6	15.2	43.8	40.0	32.9	15.1
1.0	42.0	36.7	29.8	13.0	40.3	36.9	29.7	12.7

10. Stability Analysis

Following standard practice [15, 41], we further evaluate the stability of our method over 5 random seeds. Tab. 13 reports the mean and standard deviation across five independent runs on all three benchmarks. The variances remain consistently small, and the performance gains over both baselines persist under this multi-run setting. Notably, even on the challenging C-GQA dataset, *FlowComposer* delivers stable and reproducible improvements. These results demonstrate that *FlowComposer* is not only effective but also robust to different initializations.

11. Semantic Analysis of Composer

To better understand the behavior of our *Composer* and examine whether the predicted combination coefficients carry semantic meaning, we visualize the predicted attribute and object weights $(\hat{a}, \hat{b})$ for correctly classified samples across the three datasets using the *FlowComposer* based on Troika [15]. Recall that $\hat{a}$ and $\hat{b}$ correspond to the contributions of the attribute and object velocities when composing the final composition velocity. Importantly, our intention is not to enforce that the ground-truth composition velocity must lie exactly in the span of the primitive velocities, $\mathrm{span}\{v_{\text{attr}}, v_{\text{obj}}\}$. Instead, the *Composer* aims to predict an approximate combination that produces a composition velocity pointing toward the desired semantic direction. As shown in Fig. 5, the learned coefficients indeed correlate with the visual prominence of attribute and object cues, demonstrating that the *Composer* adapts its weighting based on the evidence present in the image.

Across the visualizations, we observe clear and consistent semantic tendencies. When an attribute is visually salient, such as distinctive material patterns, strong color cues, or highly recognizable texture changes, the predicted attribute coefficient $\hat{a}$ increases. In contrast, when the attribute is abstract, weakly expressed, or visually ambiguous, the *Composer* assigns a smaller value to $\hat{a}$, relying more heavily on the object velocity. Similarly, when the object identity is visually clear and structurally well preserved, the object coefficient $\hat{b}$ becomes dominant; however, if the object shape is partially occluded, deformed,

Table 13. Performance on MIT-States [17], UT-Zappos [52], and C-GQA [39], averaged over 5 random seeds with corresponding standard deviations. * marks results taken from the respective original publications.

Method	MIT-States [17]				UT-Zappos [52]				C-GQA [39]			
	Seen	Unseen	HM	AUC	Seen	Unseen	HM	AUC	Seen	Unseen	HM	AUC
CSP* [41][ICLR23]	$46.6_{\pm 0.1}$	$49.9_{\pm 0.1}$	$36.3_{\pm 0.1}$	$19.4_{\pm 0.1}$	$64.2_{\pm 0.7}$	$66.2_{\pm 1.2}$	$46.6_{\pm 1.2}$	$33.0_{\pm 1.3}$	$28.8_{\pm 0.1}$	$26.8_{\pm 0.1}$	$20.5_{\pm 0.1}$	$6.2_{\pm 0.0}$
+FlowComposer	$48.4_{\pm 0.2}$	$50.4_{\pm 0.1}$	$37.8_{\pm 0.2}$	$20.7_{\pm 0.1}$	$66.9_{\pm 0.4}$	$68.0_{\pm 0.7}$	$51.4_{\pm 0.7}$	$38.2_{\pm 0.8}$	$29.0_{\pm 0.1}$	$31.0_{\pm 0.2}$	$23.1_{\pm 0.2}$	$7.8_{\pm 0.2}$
Troika* [15][CVPR24]	$49.0_{\pm 0.4}$	$53.0_{\pm 0.2}$	$39.3_{\pm 0.2}$	$22.1_{\pm 0.1}$	$66.8_{\pm 1.1}$	$73.8_{\pm 0.6}$	$54.6_{\pm 0.5}$	$41.7_{\pm 0.7}$	$41.0_{\pm 0.2}$	$35.7_{\pm 0.3}$	$29.7_{\pm 0.2}$	$12.4_{\pm 0.1}$
+FlowComposer	$51.7_{\pm 0.3}$	$53.1_{\pm 0.1}$	$40.2_{\pm 0.1}$	$23.4_{\pm 0.1}$	$71.5_{\pm 0.6}$	$75.0_{\pm 0.3}$	$58.8_{\pm 0.4}$	$46.7_{\pm 0.3}$	$44.8_{\pm 0.1}$	$40.6_{\pm 0.1}$	$34.1_{\pm 0.1}$	$15.9_{\pm 0.1}$

or visually degraded, the Composer reduces $\hat{b}$ and compensates through attribute evidence. These trends appear consistently across MIT-States [17], UT-Zappos [52], and C-GQA [39], despite the substantial differences in attribute style, object granularity, and visual complexity among the datasets. For example, in MIT-States [17], attributes such as `sliced` or `squished` exhibit strong, localized visual patterns, leading to larger $\hat{a}$, whereas abstract attributes like `unripe`, `clean` or `dark` produce much smaller coefficients due to their weak visual expression. Similarly, in C-GQA [39], samples with vivid attribute cues (*e.g.*, `blue coat`, `brown dog` or `painted wall`) yield higher $\hat{a}$, while cases where the attribute is subtle or occluded (*e.g.*, `blue water` `black bicycle`, or `Fried Dough`) shift the weighting toward the object, resulting in larger $\hat{b}$. For the UT-Zappos [52] dataset, the attribute typically describes material or surface appearance (*e.g.*, `leather`, `satin`, `canvas`), which is prominently expressed on the shoe surface, while the object denotes the shoe type, whose structural cues are often less visually distinctive. As a result, the attribute coefficient $\hat{a}$ consistently dominates across UT-Zappos samples, reflecting the stronger visual salience of material-based attributes.

The sample-level behaviors in Fig. 5 further highlight the interpretability of the predicted coefficients. When attribute cues are strong or highly distinctive, the Composer increases the corresponding attribute weight, whereas objects with vivid, easily recognizable visual characteristics naturally lead to larger object coefficients. For example, in the two `blue coat` samples from C-GQA [39], the first image exhibits a much more prominent blue coloration, leading to a larger attribute coefficient, whereas in the second image the coat shape is clearly visible but the blue cue is relatively weak, resulting in a higher object coefficient $\hat{b}$. Similarly, for the two `moldy tomato` samples from MIT-States [17], the upper image contains more pronounced moldy regions and multiple clearly visible tomatoes, which yields larger values for both the attribute and the object coefficients compared to the lower sample, where the mold is less salient and fewer tomatoes are present.

These observations collectively reinforce the core motivation behind our *Composer*: different composites and even different samples of the same composite exhibit varying visual contributions from the attribute and the object, so a fixed weighting scheme (as used in prior multi-branch methods such as Troika [15]) cannot adequately capture such diversity. In contrast, our learned coefficients provide sample-specific balancing of primitive velocities, allowing the model to adapt its reliance on attribute or object information according to their respective visual strengths. This semantic analysis therefore supports our design choice of learning explicit, data-dependent combination coefficients and demonstrates that the *Composer* produces weights that align with human-interpretable visual cues rather than applying uniform or heuristic mixing rules.

12. Social Impacts

This paper presents a feasible approach to compositional zero-shot learning, enabling models to recognize novel attribute–object compositions from known primitives, which may potentially benefit a variety of applications such as retrieval, assistive perception, or visual reasoning in open-world environments. However, there remains a potential risk associated with the reliance on pretrained vision–language models and benchmark datasets. In particular, our method may inherit and amplify dataset biases—such as skewed attribute distributions or culturally sensitive descriptors—which can propagate through the compositional mechanism. Such amplified biases may in turn lead to incorrect or inappropriate composition predictions in high-stakes scenarios, where misidentifying attributes or objects could result in harmful downstream decisions. Additionally, to mitigate potential negative social impacts, the development of rigorous bias-auditing tools, transparent evaluation practices, and responsible deployment protocols is crucial to ensure that such models do not propagate harmful biases or influence high-stakes decisions without proper safeguards.

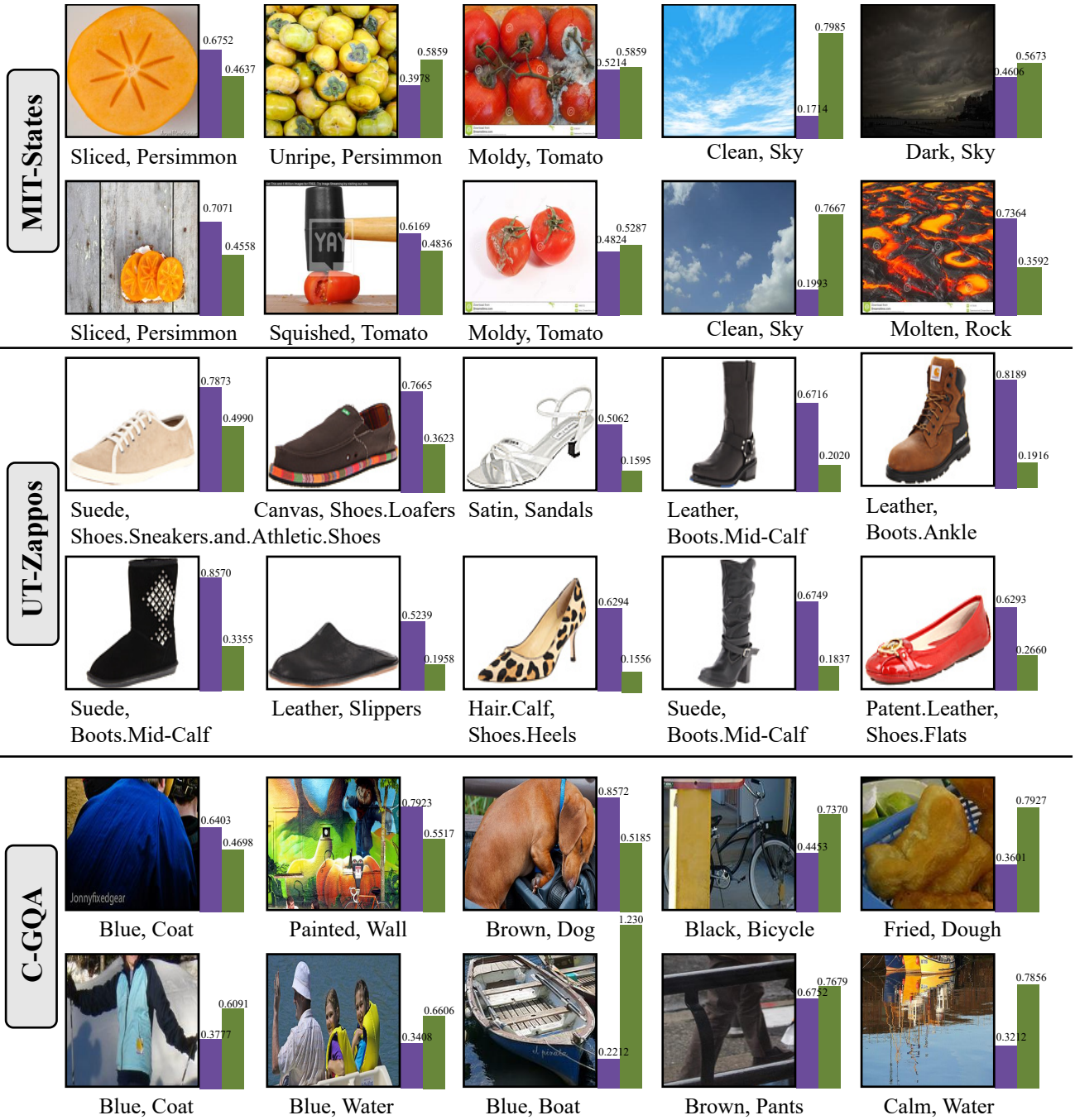

Figure 5. The purple bar denotes the value of $\hat{a}$ and the green bar denotes the value of $\hat{b}$, which are the coefficients for attribute and object velocities respectively.