COLLAGE: Adaptive Fusion-based Retrieval for Augmented Policy Learning

Sateesh Kumar, Shivin Dass, Georgios Pavlakos*, Roberto Martín-Martín*
The University of Texas at Austin
{sateesh, sdass, pavlakos, robertomm}@utexas.edu

Abstract: In this work, we study the problem of data retrieval for few-shot imitation learning: select data from a large dataset to train a performant policy for a specific task, given only a few target demonstrations. Prior methods retrieve data using a single-feature distance heuristic, assuming that the best demonstrations are those that most closely resemble the target examples in visual, semantic, or motion space. However, this approach captures only a subset of the relevant information and is prone to introducing detrimental demonstrations, e.g., retrieving data from unrelated tasks due to similar scene layouts, or selecting similar motions from tasks with divergent goals. We present COLLAGE, a method for COLLective data AGgrEgation in few-shot imitation learning that uses an adaptive late fusion mechanism to guide the selection of relevant demonstrations based on a task-specific combination of multiple cues. COLLAGE follows a simple, but flexible and efficient data aggregation recipe: it assigns weights to subsets of the dataset that are pre-selected using a single feature (e.g., appearance, shape, or language similarity), based on their task relevance, measured by how well a policy trained on each subset predicts actions in the few target demonstrations. These weights are then used during policy training to perform importance sampling over the aggregated dataset, sampling data more densely or sparsely, according to their estimated relevance. This weighted aggregation strategy is general and feature-agnostic, allowing COLLAGE to combine and leverage any number of subsets selected by any retrieval heuristic or method, and to identify which subset provides the most benefit for the target task. In extensive experiments, COLLAGE outperforms state-of-the-art retrieval and multi-task learning approaches, achieving a 5.1% improvement over the best baseline in simulation across 10 tasks, and a 16.6% improvement in the real world across 6 tasks. For our real world experiments, we include data selection from the large-scale, real-world DROID dataset, significantly improving few-shot imitation policy training. More information at our website.

Keywords: Imitation Learning, Few-Shot Learning, Data Retrieval

1 Introduction

Imitation Learning (IL) [1, 2, 3] has emerged as a powerful framework for training robot policies by mimicking expert demonstrations. With the rise of transformer- and diffusion-based models, single task IL approaches have achieved impressive performance on complex robotic manipulation domains [4, 5, 6]. However, these successes often depend on the availability of extensive demonstration data on the specific task of interest. In real-world environments, such as homes, where object configurations and task requirements frequently change, collecting sufficient demonstrations for every possible task and variation is costly, time-consuming, and ultimately impractical.

As a solution to overcome the need for task-specific demonstrations, the community has explored training generalist multi-task policies [7, 8, 9] on large-scale datasets [10, 11], which cover hundreds

* Equal Advising. Correspondence to: sateesh@utexas.edu

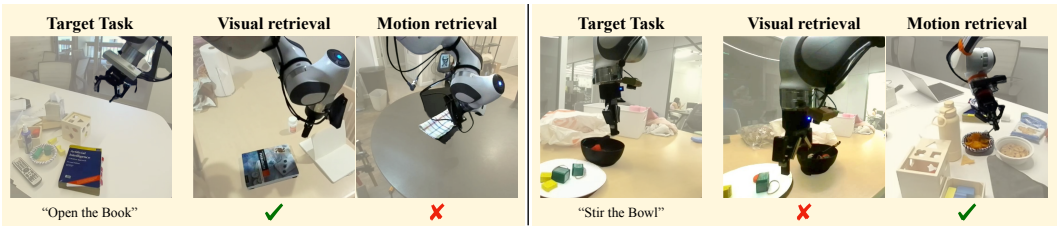

Figure 1: **Different tasks benefit from retrieval based on different modalities.** For the target task "Open the book" (left), retrieval based on visual similarity tends to return relevant demonstrations where a robot opens a book, whereas retrieval based on motion similarity often returns demonstrations with similar motions but different semantics, such as opening a cloth. On the other hand, for the target task "Stir the bowl" (right), retrieval based on motion similarity is more effective, returning demonstrations that involve stirring motions, while visual similarity tends to retrieve examples that feature a bowl but involve different actions, such as placing an object inside the bowl.

of tasks and environments and even different robot morphologies. The hope is that policies trained on a diverse array of experiences will generalize to unseen tasks with minimal adaptation. However, in practice, generalist policies often underperform on specific tasks when compared to expert policies trained on a sufficiently large number of demonstrations. This seems to be largely caused by a *negative transfer* effect [12], where irrelevant or conflicting demonstrations degrade the policy's ability to focus on task-relevant behavior.

To address this issue, a recently explored alternative for task-specific imitation policy learning leverages the same existing large demonstration datasets in a novel way: to retrieve data and augment a small dataset with a few target-task-specific demonstrations [13, 14, 15, 16]. Prior approaches in retrieval-augmented few-shot imitation learning rely on different types of similarity to retrieve relevant trajectories, including visual features [13, 15], optical flow [14], or language embeddings [17], i.e., they assume that the most helpful trajectories to train for a task are those that look, move, or are called similarly in the large dataset. However, while each of these assumptions is true in a broad, statistical sense, they can easily fail for specific cases (see Fig. 1), rendering methods relying on only one of these single-modality heuristics for retrieval highly variable and brittle. Motivated by the limitations of single-modality retrieval, we ask: *Can we design a strategy to automatically combine data retrieved using different similarity measures for higher performance in retrieval-augmented few-shot policy learning?*

We present COLLAGE, a method that demonstrates that the above is indeed possible by COLLectively AGgrEgating subsets of the dataset preselected using single-modality heuristics in a synergistic manner. For aggregation, COLLAGE proposes associating weights with each single-modality subset based on an estimate of their task-training relevance. To compute this estimation, COLLAGE employs a rollout-free mechanism: it trains reference policies on each retrieval subset and estimates the policy task-relevance by evaluating the log-likelihood of the few target demonstrations. We can consider conceptually these weights to be an approximation of the probability of a subset to be helpful to train a performant policy for the given task, and use them to guide an importance sampling mechanism during policy training: more relevant subsets of data are sampled more densely compared to less relevant subsets during the learning process. As a result, COLLAGE is agnostic to the type of retrieval features, enabling flexible integration of different similarity modalities.

We evaluate COLLAGE in both simulated and real-world settings. In simulation, we use the LIBERO benchmark [18], and demonstrate that our approach significantly outperforms both single-feature retrieval and generalist policy learning baselines by 5.1% and 14.8% respectively.

For our real-world evaluation, we retrieve from the DROID dataset [10], and demonstrate that our method improves few-shot imitation learning performance by 16.6% over existing state-of-the-art, achieving robust retrieval directly from large, diverse offline datasets without requiring manual curation, even in the presence of substantial visual and task domain shifts.

2 Related Work

COLLAGE is a novel methodology for collective data aggregation in few-shot imitation learning settings. We begin by reviewing prior work in this area, followed by related work in zero-shot manipulation and multi-modal representation learning.

Data retrieval for few-shot imitation learning methods assume access to a small number of target demonstrations that are augmented with data from a large dataset to improve policy training performance [16, 13, 14, 15, 17, 19, 20, 21, 22]. These methods employ different similarity measures to select the subset of data most similar to the target few demonstrations from a large-scale robotics dataset such as DROID [10] or the Open X-Embodiment (OXE) dataset [11]. For instance, Du et al. [13] use state-action similarity, while Lin et al. [14] use optical flow to measure motion similarity. Zha et al. [17] and Wang et al. [19] retrieve based on language similarity. Closer to our work, STRAP [15] retrieves sub-trajectories using DINO-based visual similarity [23]. COLLAGE generalizes this by performing retrieval across multiple modalities and adaptively weighting their contributions during training. During the preparation of this manuscript, a concurrent work [24] appeared that estimates an optimal demonstration subset to retrieve using a supervised, data-driven objective inspired by datamodels [25]. In contrast, COLLAGE fuses multiple subsets, each potentially retrieved using different strategies including [24], by assigning weights based on their utility for the target task

Retrieval for zero-shot Manipulation. A related body of work in robotics applies similar ideas to a zero-shot setting [26, 27, 28, 29, 30]. Using different measures of similarity with respect to the current observations, they are able to retrieve data from large datasets to develop useful manipulation policies. For example, DINO-bot [28] uses DINO features to identify the most similar demonstration in the dataset and replays its actions for task completion. Similarly, Papagiannis et al. [30] retrieve demonstrations by first applying language-based filtering, followed by visual similarity, and then track $3D$ keypoints from the selected demonstration to guide execution. Kuang et al. [29] measure feature similarity from a stable diffusion model [31] to replicate affordances from human videos, eliminating the need for a robotic dataset. While these methods do not require in-domain demonstrations, they assume that the large prior dataset of demonstrations contains trajectories that can be directly replayed to solve the target task, a strong assumption in unstructured, highly variable environments like homes.

Multi-modal representations have been widely explored in artificial perception [32, 33, 34, 35, 36] and robotics [37, 29]. Fusion methods generally follow either early or late fusion [38]. In early fusion, raw modality data is combined before processing [32, 33, 34], while in late fusion, each modality is processed independently and then combined—commonly seen in transformer architectures that fuse modality-specific tokens [35, 36]. COLLAGE follows a *late fusion* paradigm: each modality is used independently to retrieve a subset of relevant data, and these subsets are then collectively aggregated (fused) synergistically based on their relevance. In the context of data retrieval, Yu et al. [39] propose a multi-stage pipeline that filters data using features like DINO and Stable Diffusion in a fixed sequence, while Kuang et al. [29] present a modality-specific retrieval process conditioned on language, vision, and proprioception. These methods often require manually tuning the order and thresholds for each feature, limiting scalability. By contrast, COLLAGE provides a data-driven, feature-agnostic approach that easily scales to any number or type of modalities. Its fusion mechanism assigns relevance scores to each modality's retrieved subset, allowing the final result to reflect their combined utility. This weighting strategy draws from importance-based data selection, where high-utility data points—such as human interventions [40] or rare classes [41]—are emphasized. More broadly, it connects to importance sampling [42], where samples are reweighted to approximate expectations under a target distribution, enabling more effective learning from heterogeneous and imbalanced data.

3 COLLAGE

Problem Formulation. We consider a few-shot imitation learning setting where the goal is to learn a policy given a small set of expert demonstrations. Specifically, we are given a target dataset, $\mathcal{D}_{target} = \{t_1, t_2, \cdots, t_n\}$, consisting of n expert trajectories. Each trajectory t_i is a sequence of state-

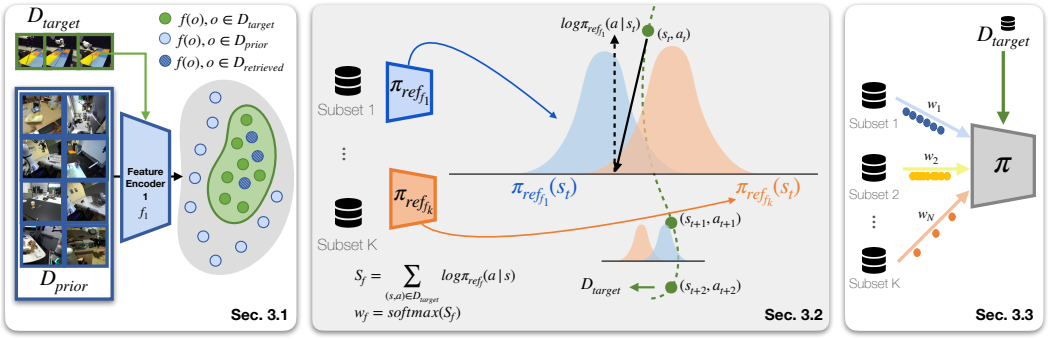

Figure 2: **Overview of our proposed COLLAGE approach.** Left: Given a set of target demonstrations $\mathcal{D}_{target}$, each modality f selects a set of retrieved trajectories $\mathcal{D}^f_{retrieved}$ from a prior dataset $\mathcal{D}_{prior}$. Center: We use the retrieved trajectories for each modality to train a reference policy π_{ref_f}. For each reference policy, we compute the log-likelihood $\log \pi_{\text{ref}_f}(a_t|s_t)$ of each state s_t of the target dataset. With this, COLLAGE estimates adaptively the importance w_f of each modality. Right: We train our final policy using the retrieved examples of each modality. We perform importance sampling, such that the sampling probability for each retrieved set is equal to w_f.

action pairs, $\{(s_1, a_1), (s_2, a_2), \cdots, (s_h, a_h)\}$, accompanied by a task instruction ℓ. Considering the limited size of $\mathcal{D}_{target}$, it is not possible to train a high-performing policy on this dataset alone. Instead, we are also given access to a large-scale offline dataset, $\mathcal{D}_{prior} = \{t^p_1, t^p_2, \cdots, t^p_m\}$, with $m \gg n$, that can support augmented policy learning. The assumption is that a policy, π_{aug}, trained via imitation learning on an augmented dataset combining $\mathcal{D}_{target}$ with the right subset of $\mathcal{D}_{prior}$ would have higher performance than training exclusively in the few demonstrations of $\mathcal{D}_{prior}$, enabling efficient few-shot imitation learning.

Overview. Our proposed approach, COLLAGE, formulates this challenge as a retrieval problem: we aim to identify demonstrations from $\mathcal{D}_{prior}$ that are most relevant to those in $\mathcal{D}_{target}$ and are therefore more likely to support effective policy learning for the target task. Rather than relying on a single-modality similarity measure, as in prior work, COLLAGE leverages multiple modalities to retrieve complementary and diverse demonstrations.

More specifically, in our setting, each state s consists of multi-modal observations $o = \{I, d, r\}$, where I is the RGB image, d is the depth map, and r is the robot state (e.g., joint angles or end-effector pose). We assume access to k feature encoders $\{\mathcal{F}_1, \mathcal{F}_2, \cdots, \mathcal{F}_k\}$, where each feature encoder $\mathcal{F}_i$ corresponds to a different modality (e.g., vision, motion, shape), and maps an observation o to a low-dimensional embedding suitable for computing similarity.

Our approach first retrieves subsets of relevant trajectories $\{\mathcal{D}^{f_1}_{retrieved}, \cdots, \mathcal{D}^{f_k}_{retrieved}\}$ by using each feature encoder to measure similarity between demonstrations in the target and prior datasets (Figure 2, left). Then, we estimate an importance weight w_f for each retrieved subset (Figure 2, center). Finally, we train a language-conditioned policy using the union of the target dataset and all retrieved datasets. During policy training, we sample trajectories following the estimated importance weights w_f. (Figure 2, right).

3.1 Retrieval Across Multiple Modalities

In this section, we describe how to obtain the retrieved set for each feature modality.

Granularity of Data Retrieval. Before retrieving relevant data, we must decide the level of granularity for retrieval: individual states, sub-trajectories, or entire trajectories from $\mathcal{D}_{prior}$. Taking inspiration from [15], we primarily adopt sub-trajectory retrieval, which enables more flexible and

semantically meaningful matching. However, our approach is agnostic to this choice, and in our instantiation of COLLAGE, we also incorporate a trajectory-level retrieval feature (see Section 3.4).

Sub-trajectory based Data Retrieval. Following Memmel et al. [15], we first segment each demonstration in $\mathcal{D}_{target}$ into sub-trajectories using an action-based heuristic based on end-effector velocity. For each feature modality, we use the corresponding encoder $\mathcal{F}_i$ and perform retrieval independently. Given a segmented target sub-trajectory t', and a trajectory t from $\mathcal{D}_{prior}$, we compute a pairwise cost matrix $C \in \mathbb{R}^{|t'|\times|t|}$ where $C_{ij} = \|\mathcal{F}_i(O_i) - \mathcal{F}_i(O_j)\|_2$. We then apply Subsequence Dynamic Time Warping (S-DTW) to align t' with a contiguous sub-sequence of t. For each target sub-trajectory, we retrieve the top-K lowest-cost matches from $\mathcal{D}_{prior}$ according to S-DTW. Readers are referred to Section A.4 and [43] for formal definitions of DTW and S-DTW.

This retrieval process is repeated separately for each feature modality, resulting in a set of modality-specific retrieval datasets $\{\mathcal{D}^{f_1}_{retrieved}, \cdots, \mathcal{D}^{f_k}_{retrieved}\}$. Each retrieved sub-trajectory also inherits its corresponding language instruction.

3.2 Estimating the Weights for Retrieved datasets

Given the modality-specific retrieved datasets $\mathcal{D}^{f_1}_{retrieved}, \cdots, \mathcal{D}^{f_k}_{retrieved}$, a simple strategy, adopted in prior work [13, 14, 15], is to uniformly combine them and perform augmented policy learning. However, as we demonstrate in our experiments, this naive uniform merging often leads to suboptimal performance. This is because, for a given task, some modalities retrieve more informative examples than others (see Figure 1).

To address this, we propose an adaptive weighting strategy that draws inspiration from importance sampling [42]. In this view, each retrieval modality f induces a proposal distribution over examples in $\mathcal{D}_{prior}$, and our goal is to sample examples from $\mathcal{D}^{f}_{retrieved}$ that reflect how well these proposals approximate the target task distribution. Concretely, for each modality f, we train a lightweight behavior cloning (BC) policy π_{ref_f} using only the modality-specific retrieved data $\mathcal{D}^{f}_{retrieved}$:

$$\pi_{\text{ref}_f} = \arg\min_{\theta} \mathcal{L}_{\text{BC}}\left(\theta; \mathcal{D}^{f}_{retrieved}\right). \tag{1}$$

Then, we evaluate the relevance of each modality by computing the log-likelihood of the target demonstrations under the corresponding reference policy:

$$S_f = \sum_{(s,a,\ell)\in\mathcal{D}_{target}} \log \pi_{\text{ref}_f}(a \mid s, \ell) \tag{2}$$

These scores are normalized using a softmax function to produce a set of modality weights: $w_f = \frac{\exp(S_f/\tau)}{\sum_{f'}\exp(S_{f'}/\tau)}$.

Intuitively, a modality that retrieves more relevant data, –i.e., one whose reference policy better explains the target task– receives a higher weight, analogous to a higher importance weight in importance sampling. The softmax normalization ensures that the weights are positive and sum to one, enabling us to automatically prioritize the most useful retrievals during training.

3.3 Retrieval Augmented Policy Learning

The next step is to train a language-conditioned visuomotor policy π using behavior cloning. The policy is a transformer-based model [44], and takes as input the past h observations $s_{t-h:t}$ along with a task instruction ℓ, predicting h future actions using a Gaussian Mixture Model (GMM) head. Following recent approaches [45, 46], our training objective combines multi-step prediction loss with ℓ_2 regularization on model parameters θ.

$$\mathcal{L}(\theta) = \mathbb{E}_{(s_{t-h:t},\, a_{t:t+h},\, \ell)\sim\mathcal{D}}\left[-\log\, \pi_\theta(a_{t:t+h} \mid s_{t-h:t},\, \ell)\right] + \lambda\|\theta\|_2^2 \tag{3}$$

Sampling Strategy. For each modality f, we construct an augmented dataset $\mathcal{D}^{f}_{\text{aug}} = \mathcal{D}^{f}_{retrieved} \cup \mathcal{D}_{target}$. During training, we sample examples from the augmented datasets $\mathcal{D}^{f}_{\text{aug}}$, using the sampling weights w_f we estimated in Section 3.2. This results in training batches containing examples retrieved by different modalities, biased toward those deemed more useful for the target task.

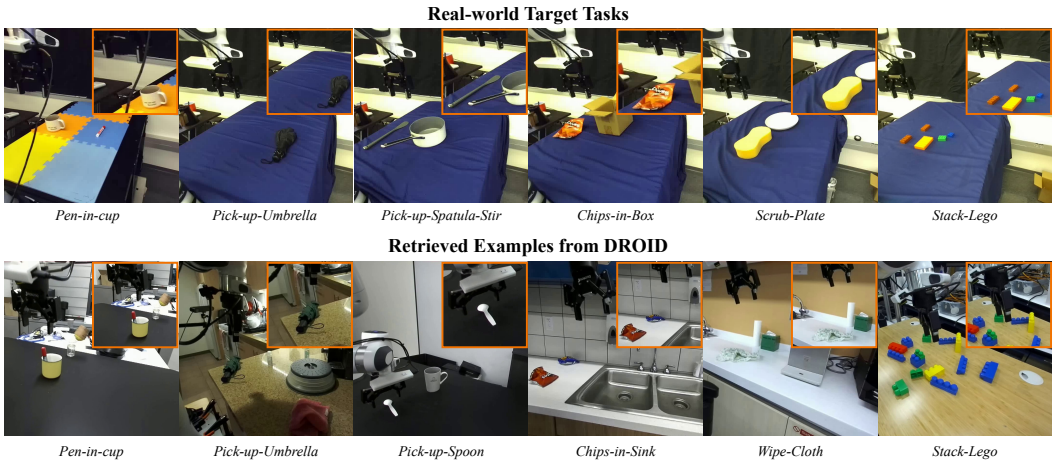

Figure 3: **Real-world tasks and corresponding retrieved examples from DROID.** Our real-world setting (top) is visually very different from DROID (bottom), yet we are able to retrieve meaningful and helpful demonstrations for each task. In many cases our target task does not align perfectly with DROID examples, but we are still able to retrieve a relevant demonstration (e.g., wipe-cloth demonstration for the related scrub-plate task), or return a partial motion of a more complicated task (e.g., pick-up-spoon is relevant for the pick-up-spatula segment of the pick-up-spatula-stir task).

3.4 Feature Modalities for Data Retrieval

We instantiate COLLAGE using four feature modalities to measure similarity between D_{target} and D_{prior}: *visual*, *motion*, *shape*, and *language*, derived from DINOv2 [23], Optical Flow [47], PointNet [48], and OpenAI embeddings [49], respectively. Below, we outline how features are extracted for a single state s_i from a demonstration d; additional implementation details are in Sections A.1 and A.2.

Visual Similarity Feature. We use DINOv2 [23] to extract visual features from the RGB image I_i of each state. The visual feature is defined as $\mathcal{F}_v(s_i) = \text{DINO}(I_i)$.

Motion Similarity Feature. We use Optical Flow [47] to measure motion similarity. Unlike DINO features, which can be extracted using a pretrained model, this modality requires a custom pretraining stage to reduce the dimensionality of raw optical flow. To this end, we first extract optical flow for the entire $\mathcal{D}_{prior}$ dataset using GMFlow [50]. The optical flow o_i is computed between each image I_i of s_i and I_{i+j} of s_{i+j}, where j is determined based on the temporal offset used in the downstream behavior cloning (BC) policy. This results in the dataset $\text{FLOW}_{\text{prior}} = \{\mathcal{O}(I_i, I_{i+j}) \mid (s_i, a_i) \in \mathcal{D}_{prior}\}$, where $\mathcal{O}$ denotes the optical flow operator, i.e., GMFlow in our case. Following Lin et al. [14], we train an encoder $p_\theta(o_i)$ and a decoder $q_\phi(\cdot)$ using a reconstruction loss defined as $L(\theta, \phi) = \mathbb{E}_{o_i \sim \text{FLOW}_{\text{prior}}} \|q_\phi(p_\theta(o_i)) - o_i\|$. The encoder-decoder model is trained on $\mathcal{D}_{prior}$, and as shown by Lin et al. [14], given a sufficiently large $\mathcal{D}_{prior}$, the learned representation generalizes well to $\mathcal{D}_{target}$. Finally, the motion feature is defined as $\mathcal{F}_m(s_i) = p_\theta(o_i)$.

Shape Similarity Feature. To represent the 3D geometry of the scene, we convert the depth map m_i into a point cloud $\mathbf{p}_i$ after background segmentation. The point cloud is then passed through a pretrained PointNet++ [48] model to compute the shape feature: $\mathcal{F}_p(s_i) = \text{PointNet}(\mathbf{p}_i)$.

Language Similarity Feature. We use OpenAI embeddings [49] to encode the instruction l_d associated with each demonstration. The language feature is computed as $\mathcal{F}_l(l_d) = \text{OpenAIEmbed}(l_d)$.

4 Experiments

We evaluate our method on the simulated LIBERO [18] benchmark. Additionally, we also test our method in the real world on 6 manipulation tasks using the DROID dataset [10]. Below, we provide a brief description of the tasks evaluated in this work which are also shown in Figure 3.

Tasks	Mug-M	Book	Cheese	Soup-Sa	Mug-Mi	Soup-C	Bowl	Stove	Mug-P	Avg.
BC	25.3	44.7	47.3	17.3	20.7	25.3	72.0	<u>73.3</u>	<u>19.3</u>	34.5
MT	31.3	**89.3**	33.3	15.3	21.0	23.3	66.7	68.0	9.3	35.8
BR [13]	32.7	38.0	33.3	29.3	16.3	28.0	74.7	63.3	6.7	32.2
FR [14]	28.7	58.0	29.3	17.3	21.3	23.3	83.3	71.3	16.7	34.9
STRAP [15]	**57.3**	85.3	29.3	16.7	29.3	**42.7**	91.3	**85.3**	18.7	45.6
Motion	48.0	66.7	60.7	25.3	28.0	30.0	48.0	<u>73.3</u>	15.3	39.5
Language	34.7	61.3	66.0	32.0	20.7	31.3	93.3	58.0	13.3	41.1
Shape	<u>54.0</u>	77.3	**68.7**	28.7	20.7	28.7	94.0	63.3	12.7	44.8
Visual [15]	40.7	66.0	66.7	30.0	<u>31.3</u>	29.3	95.3	<u>73.3</u>	**21.3**	45.4
NA-Fusion	<u>54.0</u>	<u>86.7</u>	55.3	<u>42.7</u>	24.7	28.7	94.7	67.3	14.7	46.9
COLLAGE	51.3	**89.3**	<u>67.3</u>	**53.3**	**32.7**	<u>33.3</u>	**96.0**	68.7	13.3	**50.5**

Table 1: **Performance comparison on the *LIBERO-10* benchmark.** We compare COLLAGE with various baselines (first two groups) and with ablations of our approach (third group). **Bold** indicates best performance and **<u>underline</u>** indicates second best results. All sub-trajectory method results are reported with the number of sub-trajectories, $K = 100$. Results are averaged over 3 seeds and 50 trials. The last column reports the average across all 10 tasks in *LIBERO-10*.

4.1 Experimental Setup

Tasks & Datasets. For our simulated experiments, we use 5 demonstrations from each of the 10 tasks in the *LIBERO*-10 benchmark as D_{target} and 4500 demonstrations from *LIBERO*-90 as D_{prior}.

For real-world evaluations, we use a FRANKA Panda robot. Our $\mathcal{D}_{prior}$ is a set of 30k successful episodes from the DROID dataset [10] with language instruction annotations. These episodes are selected randomly from a total of 50k successful episodes in DROID. This setting is particularly challenging, as the DROID demonstrations differ significantly in visual appearance from our robot setup (see Figure 3). Unlike previous works [13, 14, 15], we do not collect ourselves the demonstrations for $\mathcal{D}_{prior}$ and rely on automatic retrieval. We evaluate on 6 real-world tasks and collect *5* demonstrations per task. The tasks are: (1) *Pen-in-Cup*, the robot needs to pick up a pen and put it inside the cup; (2) *Umbrella*, the robot needs to pick up the umbrella; (3) *Spatula-Stir*, the robot needs to stir a pot using a spatula; (4) *Chips-Box*, the robot needs to pick up the box of chips and put it inside the box, (5) *Scrub-Plate*, where the robot scrubs a plate using a sponge and (6) *Stack-Lego*, where the robot stacks one Lego block on top of another. All these tasks require fine-grained manipulation and are hard to learn directly from 5 demonstrations. Notably, while some tasks are directly represented in DROID (e.g., *Pen-in-Cup*), others, such as *Scrub-Plate*, are not explicitly demonstrated, to the best of our knowledge. The tasks are selected to span a diverse set of behaviors, highlighting the importance of retrieving data using multiple modalities.

Baselines and Ablations. We compare COLLAGE against: **Non-retrieval methods:** (1) **BC**: training a transformer-based policy using only $\mathcal{D}_{target}$; (2) **MT**: training a multi-task transformer policy on $\mathcal{D}_{target} \cup \mathcal{D}_{prior}$; **Retrieval-based methods:** (3) **BR** [13]: cosine similarity retrieval in a VAE-based state-action embedding space trained on $\mathcal{D}_{prior}$; (4) **FR** [14]: same as BR, but with a VAE trained on optical flow (5) **STRAP** [15]: sub-trajectory retrieval using S-DTW with DINO features (numbers reported from [15]); (6) **Motion**: sub-trajectory retrieval using optical flow for motion similarity; (7) **Language**: retrieval at the demonstration level using OpenAI language embeddings. (8) **Shape**: sub-trajectory retrieval using PointNet for 3D shape similarity; (9) **Visual**: STRAP reproduced by us. **Ablation:** (10) **Non-Adaptive (NA)-Fusion**: uniform sampling across all retrieved demonstrations, instead of using per-modality adapted weights. More details in Appendix A.1 and A.2.

4.2 Results

Our experimental evaluations aim to answer the following questions:

Method	Pen	Umb	Stir	Chips	Lego	Scrub	Avg.
BC	0/15	2/15	1/15	3/15	0/15	0/15	6.7
MT	0/15	0/15	0/15	1/15	2/15	0/15	3.3
Visual [15]	4/15	4/15	3/15	**9/15**	5/15	1/15	28.9
Motion	0/15	0/15	4/15	**9/15**	1/15	1/15	16.7
Shape	0/15	3/15	3/15	6/15	**6/15**	0/15	20.0
Language	4/15	4/15	5/15	4/15	2/15	6/15	27.7
COLLAGE	**6/15**	**6/15**	**9/15**	7/15	**6/15**	**7/15**	**45.5**

Table 2: **Real-world results.** We perform 15 trials for each method and we report the number of successful episodes for each task. The last column refers to the average percentage of success.

Task	w^{Visual}	w^{Motion}	w^{Shape}	w^{Language}
LIBERO-10				
Cheese	0.28	0.18	0.46	0.07
Soup-C	0.11	0.52	0.22	0.14
DROID				
Lego	0.6	0.02	0.28	0.1
Pen	0.30	0.24	0.10	0.36

Table 3: **Importance weights for various tasks.** We report the weights for each modality for simulation (LIBERO-10) and real-world (DROID) tasks.

(1) *Can we improve imitation learning performance by combining multiple feature modalities?* Our evaluation on the *LIBERO-10* benchmark (Table 1) shows that COLLAGE improves average performance by 5.1% and 5.7% over *Visual* and *Shape* based retrieval respectively. While *Visual* is the strongest single-modality baseline overall, it often fails to retrieve demonstrations in cases with significant appearance changes. For example, on the *Book* task, *Visual* modality based retrieval does not return demonstrations in $\mathcal{D}_{prior}$ that share the same instruction but differ in the visual appearance of the scene. In contrast, COLLAGE retrieves such demonstrations since it also relies on the *Motion* and *Language* modalities, resulting in a 23.3% improvement over *Visual* modality alone. Similarly, *Motion*-based retrieval underperforms on many tasks because it ignores semantic information which is critical for some tasks. Across the benchmark, COLLAGE also consistently outperforms prior retrieval baselines, such as *BR* and *FR*, in imitation learning performance. We report results for $9/10$ tasks and note that for the *Moka-Moka* task, all approaches manage success rate of 0 and is not included in the table.

(2) *Can* COLLAGE *effectively retrieve relevant demonstrations from large-scale, diverse datasets that have minimal overlap to the target tasks?* Prior works [13, 14, 15] demonstrate data retrieval by collecting a set of $\mathcal{D}_{prior}$ demonstrations that are relatively similar to the target demonstrations. In contrast to that, we go one step further and tackle the challenging setting of using as $\mathcal{D}_{prior}$ an independent large-scale dataset, *DROID*. From our experiments (Table 2), we observe that *BC* performs poorly in the real-world setting, with just $0.5/15$ successes on average–highlighting the difficulty of learning fine-grained behaviors from limited demonstrations. Individual modalities perform well on specific tasks (e.g., *Language* on *Scrub-Plate*, *Shape* on *Stack-Lego*, *Visual* on *Chips-Box*), but COLLAGE consistently outperforms all of them. We find that adaptively sampling the per-modality demonstrations improves robustness to real-world variations such as lighting and object pose. Interestingly, COLLAGE can outperform the best individual modalities by combining complementary behaviors across modalities (e.g. *Spatula-Stir*, COLLAGE $9/15$ vs *Language* $5/15$) .

(3) *How important is adaptive weighting when using data retrieved from multiple modalities?* As shown in Table 1, COLLAGE outperforms the non-adaptive baseline (NA-Fusion) across all tasks (50.5 vs. 46.9 average success rate). We find that the importance weights predicted by COLLAGE (Table 3) are high for modalities with high success rates (e.g., *Shape* on *Cheese-Butter* and *Motion* on *Soup-Cheese*). Similarly, for real-world tasks, COLLAGE down-weights the less informative modality (e.g., *Motion* for Lego and *Shape* for Pen-in-Cup).

5 Conclusion

We propose COLLAGE, an approach that enables robust retrieval-augmented policy learning by adaptively combining data retrieved using different similarity measures. Unlike prior approaches that rely on fixed feature types or manually designed retrieval pipelines, COLLAGE is agnostic to the choice of retrieval modality, making it broadly applicable across diverse settings. Our results show that adaptively weighting data retrieved from diverse similarity modalities improves few-shot imitation learning performance, both in simulation and when retrieving from large-scale, visually diverse datasets like DROID. We will publicly release our code to support further research.

6 Limitations

While COLLAGE provides a flexible and general approach for combining data retrieved via multiple modalities, it has a few practical limitations. The performance of our method ultimately depends on the quality of the retrieved data—if all modalities fail to retrieve relevant examples, performance gains may be limited. However, we find this to be rare in practice, especially when using diverse modalities that offer complementary strengths. Additionally, our approach requires training a reference policy for each modality, which, although lightweight, introduces additional computational overhead.

An exciting direction for future work is to explore more efficient ways to approximate these reference policies or develop proxy metrics that can predict the utility of retrieved data without full policy training.

Acknowledgments

We thank Jiaheng Hu and Albert Yu for their valuable feedback on the manuscript, and appreciate fruitful discussions with Arpit Bahety, Rutav Shah, and Sanjay Haresh. G.P. acknowledges support by NSF IIS-2504906, and Gifts from Adobe and Google. R.M.M. acknowledges support by DARPA TIAMAT HR0011-24-9-0428

References

[1] S. Schaal, A. Ijspeert, and A. Billard. Computational approaches to motor learning by imitation. *Philosophical Transactions of the Royal Society of London. Series B: Biological Sciences*, 358 (1431):537–547, 2003.

[2] B. D. Argall, S. Chernova, M. Veloso, and B. Browning. A survey of robot learning from demonstration. *Robotics and autonomous systems*, 57(5):469–483, 2009.

[3] T. Zhang, Z. McCarthy, O. Jow, D. Lee, X. Chen, K. Goldberg, and P. Abbeel. Deep imitation learning for complex manipulation tasks from virtual reality teleoperation. In *2018 IEEE international conference on robotics and automation (ICRA)*, pages 5628–5635. Ieee, 2018.

[4] T. Z. Zhao, J. Tompson, D. Driess, P. Florence, K. Ghasemipour, C. Finn, and A. Wahid. Aloha unleashed: A simple recipe for robot dexterity. *arXiv preprint arXiv:2410.13126*, 2024.

[5] C. Chi, Z. Xu, S. Feng, E. Cousineau, Y. Du, B. Burchfiel, R. Tedrake, and S. Song. Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*, page 02783649241273668, 2023.

[6] Z. Fu, T. Z. Zhao, and C. Finn. Mobile aloha: Learning bimanual mobile manipulation with low-cost whole-body teleoperation. *arXiv preprint arXiv:2401.02117*, 2024.

[7] S. Reed, K. Zolna, E. Parisotto, S. G. Colmenarejo, A. Novikov, G. Barth-Maron, M. Gimenez, Y. Sulsky, J. Kay, J. T. Springenberg, et al. A generalist agent. *arXiv preprint arXiv:2205.06175*, 2022.

[8] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, J. Dabis, C. Finn, K. Gopalakrishnan, K. Hausman, A. Herzog, J. Hsu, et al. Rt-1: Robotics transformer for real-world control at scale. *arXiv preprint arXiv:2212.06817*, 2022.

[9] K. Black, N. Brown, D. Driess, A. Esmail, M. Equi, C. Finn, N. Fusai, L. Groom, K. Hausman, B. Ichter, et al. π_0: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024.

[10] A. Khazatsky, K. Pertsch, S. Nair, A. Balakrishna, S. Dasari, S. Karamcheti, S. Nasiriany, M. K. Srirama, L. Y. Chen, K. Ellis, et al. Droid: A large-scale in-the-wild robot manipulation dataset. *arXiv preprint arXiv:2403.12945*, 2024.

[11] A. O'Neill, A. Rehman, A. Maddukuri, A. Gupta, A. Padalkar, A. Lee, A. Pooley, A. Gupta, A. Mandlekar, A. Jain, et al. Open x-embodiment: Robotic learning datasets and rt-x models: Open x-embodiment collaboration 0. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6892–6903. IEEE, 2024.

[12] Z. Wang, Z. Dai, B. Póczos, and J. Carbonell. Characterizing and avoiding negative transfer. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11293–11302, 2019.

[13] M. Du, S. Nair, D. Sadigh, and C. Finn. Behavior retrieval: Few-shot imitation learning by querying unlabeled datasets. *arXiv preprint arXiv:2304.08742*, 2023.

[14] L.-H. Lin, Y. Cui, A. Xie, T. Hua, and D. Sadigh. Flowretrieval: Flow-guided data retrieval for few-shot imitation learning. In *8th Annual Conference on Robot Learning*, 2024.

[15] M. Memmel, J. Berg, B. Chen, A. Gupta, and J. Francis. Strap: Robot sub-trajectory retrieval for augmented policy learning. In *The Thirteenth International Conference on Learning Representations*, 2025.

[16] S. Nasiriany, T. Gao, A. Mandlekar, and Y. Zhu. Learning and retrieval from prior data for skill-based imitation learning. In *6th Annual Conference on Robot Learning*, 2022.

[17] L. Zha, Y. Cui, L.-H. Lin, M. Kwon, M. G. Arenas, A. Zeng, F. Xia, and D. Sadigh. Distilling and retrieving generalizable knowledge for robot manipulation via language corrections. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 15172–15179. IEEE, 2024.

[18] B. Liu, Y. Zhu, C. Gao, Y. Feng, Q. Liu, Y. Zhu, and P. Stone. Libero: Benchmarking knowledge transfer for lifelong robot learning. *Advances in Neural Information Processing Systems*, 36: 44776–44791, 2023.

[19] G. Wang, Y. Xie, Y. Jiang, A. Mandlekar, C. Xiao, Y. Zhu, L. Fan, and A. Anandkumar. Voyager: An open-ended embodied agent with large language models, 2023. *URL https://arxiv.org/abs/2305.16291*, 2023.

[20] Z.-H. Yin and P. Abbeel. Offline imitation learning through graph search and retrieval. *arXiv preprint arXiv:2407.15403*, 2024.

[21] Y. Zhu, Z. Ou, X. Mou, and J. Tang. Retrieval-augmented embodied agents. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17985–17995, 2024.

[22] Y. Guo, B. Tang, I. Akinola, D. Fox, A. Gupta, and Y. Narang. Srsa: Skill retrieval and adaptation for robotic assembly tasks. *arXiv preprint arXiv:2503.04538*, 2025.

[23] M. Oquab, T. Darcet, T. Moutakanni, H. Vo, M. Szafraniec, V. Khalidov, P. Fernandez, D. Haziza, F. Massa, A. El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023.

[24] S. Dass, A. Khaddaj, L. Engstrom, A. Madry, A. Ilyas, and R. Martín-Martín. Datamil: Selecting data for robot imitation learning with datamodels. *arXiv preprint arXiv:2505.09603*, 2025.

[25] A. Ilyas, S. M. Park, L. Engstrom, G. Leclerc, and A. Madry. Datamodels: Predicting predictions from training data. *arXiv preprint arXiv:2202.00622*, 2022.

[26] S. Izquierdo, M. Argus, and T. Brox. Conditional visual servoing for multi-step tasks. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2190–2196. IEEE, 2022.

[27] F. Malato, F. Leopold, A. Melnik, and V. Hautamäki. Zero-shot imitation policy via search in demonstration dataset. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7590–7594. IEEE, 2024.

[28] N. Di Palo and E. Johns. Dinobot: Robot manipulation via retrieval and alignment with vision foundation models. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2798–2805. IEEE, 2024.

[29] Y. Kuang, J. Ye, H. Geng, J. Mao, C. Deng, L. Guibas, H. Wang, and Y. Wang. Ram: Retrieval-based affordance transfer for generalizable zero-shot robotic manipulation. *arXiv preprint arXiv:2407.04689*, 2024.

[30] G. Papagiannis, N. Di Palo, P. Vitiello, and E. Johns. R+ x: Retrieval and execution from everyday human videos. *arXiv preprint arXiv:2407.12957*, 2024.

[31] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022.

[32] G. Barnum, S. Talukder, and Y. Yue. On the benefits of early fusion in multimodal representation learning. *arXiv preprint arXiv:2011.07191*, 2020.

[33] L. Spinello and K. O. Arras. Leveraging rgb-d data: Adaptive fusion and domain adaptation for object detection. In *2012 IEEE International Conference on Robotics and Automation*, pages 4469–4474. IEEE, 2012.

[34] S. Mo and P. Morgado. Unveiling the power of audio-visual early fusion transformers with dense interactions through masked modeling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 27186–27196, 2024.

[35] X. V. Lin, A. Shrivastava, L. Luo, S. Iyer, M. Lewis, G. Ghosh, L. Zettlemoyer, and A. Aghajanyan. Moma: Efficient early-fusion pre-training with mixture of modality-aware experts. *arXiv preprint arXiv:2407.21770*, 2024.

[36] A. Nagrani, S. Yang, A. Arnab, A. Jansen, C. Schmid, and C. Sun. Attention bottlenecks for multimodal fusion. *Advances in neural information processing systems*, 34:14200–14213, 2021.

[37] R. Shah, R. Martín-Martín, and Y. Zhu. Mutex: Learning unified policies from multimodal task specifications. *arXiv preprint arXiv:2309.14320*, 2023.

[38] J. Sanchez-Riera, K.-L. Hua, Y.-S. Hsiao, T. Lim, S. C. Hidayati, and W.-H. Cheng. A comparative study of data fusion for rgb-d based visual recognition. *Pattern Recognition Letters*, 73:1–6, 2016.

[39] H. Yu, Y. Tian, S. Kumar, L. Yang, and H. Wang. The devil is in the details: A deep dive into the rabbit hole of data filtering. *arXiv preprint arXiv:2309.15954*, 2023.

[40] A. Mandlekar, D. Xu, R. Martín-Martín, Y. Zhu, L. Fei-Fei, and S. Savarese. Human-in-the-loop imitation learning using remote teleoperation. *arXiv preprint arXiv:2012.06733*, 2020.

[41] X. Chen, Y. Zhou, D. Wu, C. Yang, B. Li, Q. Hu, and W. Wang. Area: adaptive reweighting via effective area for long-tailed classification. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 19277–19287, 2023.

[42] S. T. Tokdar and R. E. Kass. Importance sampling: a review. *Wiley Interdisciplinary Reviews: Computational Statistics*, 2(1):54–60, 2010.

[43] T. Giorgino. Computing and visualizing dynamic time warping alignments in r: the dtw package. *Journal of statistical Software*, 31:1–24, 2009.

[44] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[45] S. Haldar, Z. Peng, and L. Pinto. Baku: An efficient transformer for multi-task policy learning. *arXiv preprint arXiv:2406.07539*, 2024.

[46] S. Nasiriany, A. Maddukuri, L. Zhang, A. Parikh, A. Lo, A. Joshi, A. Mandlekar, and Y. Zhu. Robocasa: Large-scale simulation of everyday tasks for generalist robots. *arXiv preprint arXiv:2406.02523*, 2024.

[47] S. S. Beauchemin and J. L. Barron. The computation of optical flow. *ACM computing surveys (CSUR)*, 27(3):433–466, 1995.

[48] C. R. Qi, L. Yi, H. Su, and L. J. Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural information processing systems*, 30, 2017.

[49] OpenAI. text-similarity-ada-001. https://platform.openai.com/docs/guides/embeddings, 2023. OpenAI language similarity embedding model.

[50] H. Xu, J. Zhang, J. Cai, H. Rezatofighi, and D. Tao. Gmflow: Learning optical flow via global matching. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8121–8130, 2022.

[51] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, J. Davison, S. Shleifer, P. von Platen, C. Ma, Y. Jernite, J. Plu, C. Xu, T. L. Scao, S. Gugger, M. Drame, Q. Lhoest, and A. M. Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online, Oct. 2020. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/2020.emnlp-demos.6.

[52] A. Mandlekar, D. Xu, J. Wong, S. Nasiriany, C. Wang, R. Kulkarni, L. Fei-Fei, S. Savarese, Y. Zhu, and R. Martín-Martín. What matters in learning from offline human demonstrations for robot manipulation. *arXiv preprint arXiv:2108.03298*, 2021.

[53] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1912–1920, 2015.

[54] J. H. Cho and P. Krähenbühl. Language-conditioned detection transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16593–16603, 2024.

A Appendix

A.1 Details of Simulated Experiments

Full Task Names for Table 1 in the main paper Below are the target abbreviations used in Table 1 and Table 5 respectively and the corresponding task name as used in *LIBERO-10* benchmark:

Abbr. (Tab. 1)	Abbr. (Tab. 5, Fig. 4)	Task Name in LIBERO-10
Mug-M	Mug-Mug	LIVING ROOM SCENE5 put the white mug on the left plate and put the yellow and white mug on the right plate
Book	Book-Caddy	STUDY SCENE1 pick up the book and place it in the back compartment of the caddy
Cheese	Cheese-Butter	LIVING ROOM SCENE2 put both the cream cheese box and the butter in the basket
Soup-Sa	Soup-Sauce	LIVING ROOM SCENE2 put both the alphabet soup and the tomato sauce in the basket
Soup-C	Soup-Cheese	LIVING ROOM SCENE1 put both the alphabet soup and the cream cheese box in the basket
Stove-Mo	Stove-Moka	KITCHEN SCENE3 turn on the stove and put the moka pot on it
Mug-Mi	Mug-Micro	KITCHEN SCENE6 put the yellow and white mug in the microwave and close it
Mug-P	Mug-Pudding	LIVING ROOM SCENE6 put the white mug on the plate and put the chocolate pudding to the right of the plate
Bowl	Bowl-Cabinet	KITCHEN SCENE4 put the black bowl in the bottom drawer of the cabinet and close it

Table 4: Mapping between abbreviations in Tables 1 and 5 and the full task names.

Retrieval Implementation Details We perform retrieval using 4 feature modalities i.e. Visual using DINO features [23], Shape using POINTNET features [48], Motion using Optical Flow features [50] and Language using embeddings obtained using OPENAI's API [49]. We set the number of retrieved sub-trajectories, *K* to 100, for DINO, PointNet and FLOW respectively. We use the agent-centric images for all modalities. Below, we describe the implementation pipeline of each feature in detail:

- **DINO**: We use the 128×128 images provided in the *LIBERO-10* benchmark and extract embeddings using *DINOv2*, specifically we use the DINOv2 model from the *transformers* codebase developed by *HuggingFace* [51]. The model outputs patch-wise low-demensional embeddings along with the *CLS* token, we apply average pooling over both to obtain a 1×768 vector for each image frame.
- **FLOW**: We extract optical flow using the GMFlow model [50], with a temporal offset of $j = 5$. To train an optical flow reconstruction model, we design a hybrid architecture that combines convolutional layers with a transformer-based encoder-decoder. Given a 128×128 flow input, we first apply a series of convolutional operations to reduce the spatial dimensions and obtain a compact token of size 1×512. This token is passed through a transformer encoder, yielding a latent embedding of size 256. The embedding is then processed by a transformer decoder and a sequence of convolutional upsampling layers to reconstruct the original optical flow, supervised using a reconstruction loss.
- **PointNet**: We first extract the depth maps using the *Robomimic*[52] codebase. Then, we use ground-truth segmentation masks obtained through the simulator to extract the foreground segmentation mask. This foreground segmentation mask is used to create the pointcloud of the scene. The pointcloud is passed through a PointNet++ model [48] to extract a 1×256 embedding vector for each observation. We use the PointNet++ model trained on the task of object classification using the *ModelNet40* [53] dataset.
- **LANG**: We implement a language-based retrieval pipeline that leverages semantic similarity between task instructions. For each task, we compute a language embedding of the instruction using OpenAI's `text-embedding-ada-002` model and compare it to embeddings from a large prior dataset using cosine similarity. Demonstrations with instruction similarity above 0.90 are selected as relevant. To ensure the number of retrieved frames matches that of other modalities, we first compute the total number of non-target frames retrieved by the baseline (e.g., visual) modality, and then distribute this frame budget evenly across the selected language-retrieved demonstrations. This enables a controlled comparison of retrieval performance across modalities under equal data budgets.

Implementation Details of Segmenting $\mathcal{D}_{target}$. To perform sub-trajectory retrieval, we segment the demonstrations in $\mathcal{D}_{target}$ using the heuristic proposed in [15]. Specifically, we implement a

trajectory segmentation heuristic based on the velocity magnitude of end-effector states. Given a sequence of 3D positions, we compute the frame-to-frame difference in the (X, Y, Z) coordinates and sum the absolute values to approximate the velocity magnitude at each timestep. Points where this velocity falls below a small threshold ϵ (set to 5×10^{-3} in simulated experiments) are interpreted as pauses in movement. The trajectory is then segmented at these pause points, producing a sequence of sub-trajectories that correspond to continuous motion segments between stops. This enables the isolation of meaningful motion primitives from longer trajectories. In practice, this strategy may lead to over-segmentation due to stop-motions in demonstrations, we fix this by merging all sub-trajectories with length greater than 20.

Baseline Implementation Details. For BR [13] and FR [14], we use the official codebase [1] from FlowRetrieval [14] to train VAEs and compute similarities. We retrieve single state action pairs and pad them by retrieving states from $t - h$ to $h - 1$ to ensure compatibility with the transformer based policy used in our experiments. For STRAP, we use the officially released codebases for retrieval [2] and policy learning [3] respectively.

Policy Training Implementation Details. All policies are trained with both agent and in-hand observation. We use the Robomimic codebase[4] to train our policies. Note that we keep our policy training and retrieval framework consistent with Memmel et al. [15]. However, we find that their reported numbers (indicated as *STRAP* in Table 1 and Table 5) do not exactly match the results we reproduced (indicated as *DINO*). That said, the average success rate across all tasks is consistent with their reported value. After discussion with the authors of Memmel et al. [15], we attribute this discrepancy primarily to the inherent stochasticity in the *Robomimic* codebase.

Hyperparameters. For our implementation as well as baselines, we train a transformer based policy is trained for a total of $60k$ steps with a batch size of 32. The number of target demonstrations is 5 in all our experiments. The temperature parameter $\mathcal{T}$ introduced in Section 3 is set to 2 for all experiments in simulation. All reported success rates are averaged over 3 seeds (1234, 42, 4325).

Results with Standard Deviations. Table 5 presents our complete results with standard deviations.

Comparison with higher values of retrieved sub-trajectories (K) For COLLAGE, we retrieve $K = 100$ sub-trajectories per modality across four modalities, resulting in $4 \times 100 = 400$ retrieved examples—compared to the 100 sub-trajectories used by a single-modality baseline. To isolate the effect of multi-modal information, we also ran an experiment retrieving $K = 400$ samples using only the best-performing modality, *DINO*, matching our total retrieval count. The total number of sequences used on average by each method in this experiment is approximately $25k$. As shown in Table 6, even with near identical retrieval size, COLLAGE outperforms the *DINO*-only setup, demonstrating that combining multiple modalities yields benefits beyond merely increasing the number of examples.

Weights Predicted by COLLAGE. Figure 4 presents the weights predicted for each modality on all tasks from the *LIBERO-10* benchmark.

Distribution of Retrieved Data from Different Modalities. In Figure 5, we visualize the demonstrations retrieved for the *Book* (top) and *Mug-Microwave* (bottom) tasks using the different modalities considered in this work. For the *Book* task, visual retrieval primarily returns demonstrations from the same scene (*scene1*) but with a different goal (e.g., placing the book in the front compartment). Because no retrieved example matches both the target scene and goal, motion and language retrieval contribute demonstrations with the correct goal but from a different scene (*scene2*). Fusing these complementary modalities enables COLLAGE to outperform any individual modality, as shown in Table 1. For the *Mug-Microwave* task, visual and shape retrieval yield demonstrations with relevant objects (e.g., yellow mug, microwave) and sub-tasks. In contrast, language retrieval surfaces demon-

[1] https://github.com/lihenglin/bridge_training_code
[2] https://github.com/WEIRDLabUW/STRAP
[3] https://github.com/WEIRDLabUW/robomimic_strap
[4] https://github.com/ARISE-Initiative/robomimic/tree/robocasa

	Mug-Mug	Book-Caddy	Cheese-Butter	Soup-Sauce	Mug-Micro
BC	25.33 ± 3.40	44.67 ± 3.40	47.33 ± 7.54	17.33 ± 2.49	20.67 ± 6.18
MT	31.33 ± 3.06	89.33 ± 5.89	33.33 ± 5.54	15.33 ± 3.46	23.33 ± 6.48
BR [13]	32.67 ± 1.15	38.00 ± 2.0	33.33 ± 1.15	29.33 ± 3.06	16.33 ± 3.27
FR[14]	28.67 ± 4.19	58.00 ± 3.27	29.33 ± 4.62	17.33 ± 3.40	21.33 ± 3.06
STRAP [15]	57.33 ± 7.7	85.33 ± 2.8	29.33 ± 11.3	16.67 ± 2.0	29.33 ± 2.7
Motion	48.00 ± 4.24	66.67 ± 20.8	60.67 ± 2.86	25.33 ± 4.11	28.00 ± 9.93
Language	34.67 ± 3.40	61.33 ± 6.60	66.00 ± 9.09	32.00 ± 8.48	20.67 ± 1.89
Shape	54.00 ± 5.89	77.33 ± 2.49	68.67 ± 15.43	28.67 ± 5.74	20.67 ± 4.71
Visual [15]	40.67 ± 3.06	66.00 ± 3.46	66.67 ± 8.08	30.00 ± 3.46	31.33 ± 7.57
NA-Fusion	54.00 ± 5.66	86.67 ± 4.99	55.33 ± 5.25	42.67 ± 5.25	24.67 ± 2.49
COLLAGE	51.33 ± 9.8	89.33 ± 4.99	67.33 ± 10.49	53.33 ± 8.38	32.67 ± 5.30

	Soup-Cheese	Moka-Moka	Bowl-Cabinet	Stove-Moka	Mug-Pudding
BC	25.33 ± 3.39	00.00 ± 0.00	72.00 ± 2.82	73.33 ± 2.49	19.33 ± 5.73
MT	23.33 ± 3.06	00.00 ± 0.00	66.67 ± 4.19	68.00 ± 3.27	9.33 ± 3.06
BR [13]	28.00 ± 4.62	00.00 ± 0.00	74.67 ± 3.77	63.33 ± 3.06	6.67 ± 3.40
FR [14]	23.33 ± 4.19	00.00 ± 0.00	83.33 ± 3.06	71.33 ± 3.40	16.67 ± 3.06
STRAP [15]	42.67 ± 7.2	00.00 ± 0.00	91.33 ± 2.2	85.33 ± 2.2	18.67 ± 1.4
Motion	30.00 ± 7.11	00.00 ± 0.00	48.00 ± 4.24	73.33 ± 3.00	15.33 ± 3.4
Language	31.33 ± 7.72	00.00 ± 0.00	93.33 ± 3.4	58.00 ± 13.95	13.33 ± 2.49
Shape	28.67 ± 5.24	00.00 ± 0.00	94.00 ± 1.63	63.33 ± 19.48	12.67 ± 9.43
Visual [15]	29.33 ± 4.6	00.00 ± 0.00	95.33 ± 2.49	73.33 ± 3.26	21.33 ± 2.49
NA-Fusion	28.67 ± 5.73	00.00 ± 0.00	94.67 ± 4.11	67.33 ± 19.96	14.67 ± 4.11
COLLAGE	33.33 ± 3.39	00.00 ± 0.00	96.00 ± 2.62	68.67 ± 16.36	13.33 ± 5.73

Table 5: Performance across 10 *LIBERO-10* tasks with standard deviation reported across 3 seeds.

Tasks	Mug-M	Book	Cheese	Soup-Sa	Mug-Mi	Soup-C	Bowl	Stove-Mo	Mug-P	Avg.
Visual ($K = 400$)	34.7	64.7	**72.0**	50.7	26	28.7	95.3	**74.7**	**16.7**	46.9
COLLAGE	**51.3**	**89.3**	67.3	**53.3**	**32.7**	**33.3**	**96.0**	68.7	13.3	**50.5**

Table 6: Comparison between *Visual* similarity only and COLLAGE while using identical retrieval set size on the *LIBERO-10* benchmark. Both methods retrieve around $25k$ sequences with number of retrieved sub-trajectories, $K = 400$. Results averaged over 3 seeds. Best result in **bold**.

strations with semantically similar instructions but often from unrelated scenes or involving different objects. Accordingly, the modality weights predicted by COLLAGE assign a low weight (0.05) to *Language* (see Figure 4), reflecting the limited task relevance of the retrieved examples.

A.2 Details of Real World Experiments

Retrieval Implementation Details Our real-world implementation follows the same pipeline described in Section A.1 for retrieving data with the *DINO*, *FLOW*, and *LANG* modalities. For the *PointNet* modality, however, we no longer have simulator-provided ground-truth depth and masks. Instead, we start from the raw video files released with DROID [10], which were captured using ZED stereo cameras. We use ZED's Python API[5] to extract per-frame depth maps. Next, we run DECOLA [54] to obtain object masks, which we combine with the depth maps to produce a foreground point cloud for each observation. Finally, this point cloud is fed into the PointNet++ model exactly as in Section A.1.

[5] https://github.com/stereolabs/zed-python-api

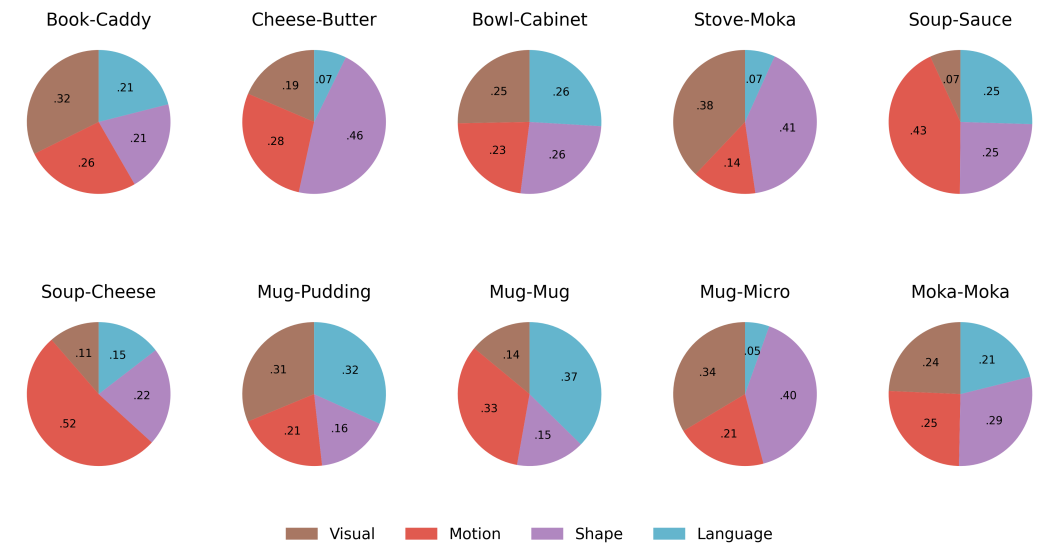

Figure 4: Modality importance weights predicted by COLLAGE for all tasks in the *LIBERO-10* benchmark.

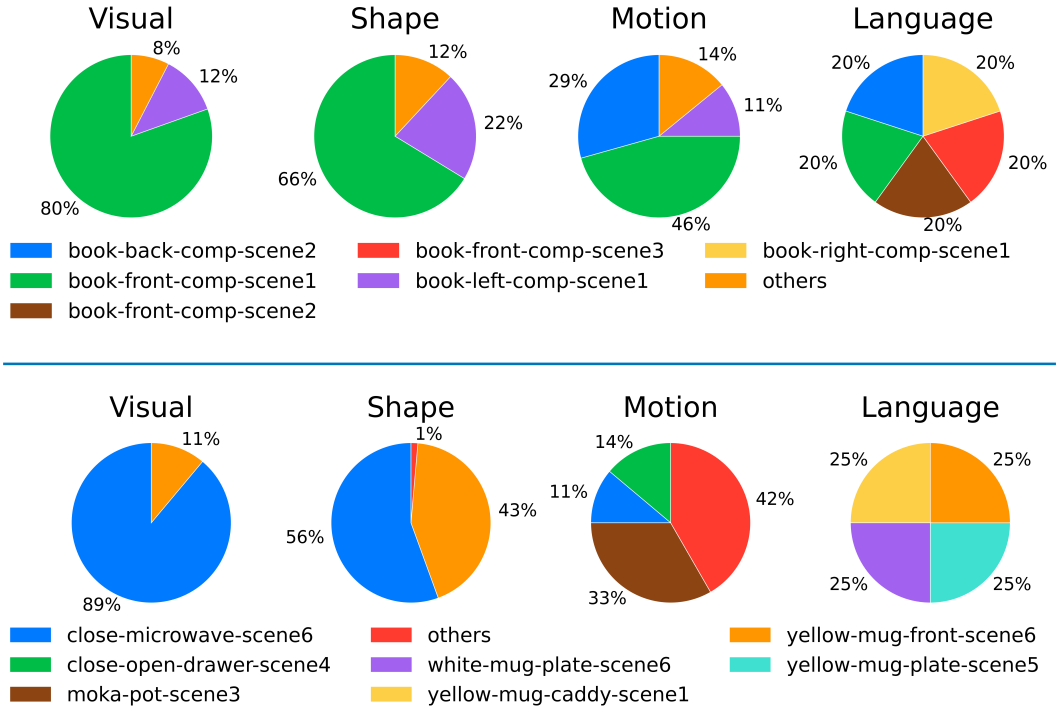

Figure 5: Visualization of the types of demonstrations retrieved from *LIBERO-90* for two *LIBERO-10* target tasks: *book-back-comp-scene1* (top) and *mug-microwave-scene6* (bottom). Each pie chart corresponds to a retrieval modality and shows the proportion of retrieved sub-trajectories originating from different *LIBERO-90* tasks. Tasks contributing less than 10% of the retrieved data are grouped under "others".

Camera Views. We use all three available camera views in DROID for retrieval. For each modality, we obtain embeddings for each camera view and the embedding for a given observation is computed as the average of the embeddings from all three viewpoints.

Hyperparameters. The temperature parameter, $\mathcal{T}$ in real world experiments is set to 10. The high value of temperature when compared to simulation experiments is mainly due to the large variance between DROID [10] data and the target demonstrations. The number of sub-trajectories retrieved, K, is set to 100 in all real-world experiments. The threshold for segmenting trajectories, ϵ, is set to 2×10^{-3}.

Policy Training Implementation Details. All baselines and our method is trained for $50k$ steps and the policies use all three camera views. The remaining policy implementation details match with those described for simulated experiments in Section A.1. We use the setup from DROID for data collection and policy rollouts [10].

A.3 Weight Estimation Implementation Details

We train reference policies using a transformer-based architecture following the objective in Eq. 3. Each policy is trained for 100 epochs (approximately one-third the rollout training time). To compute the log-likelihood score for modality f, we first define the per-epoch evaluation:

$$S_f^{(e)} = \sum_{(s,a,\ell)\in\mathcal{D}_{\text{target}}} \log \pi_{\text{ref}_f}^{(e)}\big(a \mid s, \ell\big). \tag{4}$$

We evaluate $S_f^{(e)}$ every 10 epochs, excluding the first 50 epochs (i.e. at $e \in \{60, 70, 80, 90, 100\}$). The final modality score S_f is the average over these five checkpoints:

$$S_f = \frac{1}{5} \sum_{e\in\{60,70,80,90,100\}} S_f^{(e)}. \tag{5}$$

We exclude the scores computed in the first 50 epochs because they exhibit high variance. We are interested in how well the retrieved data describes $\mathcal{D}_{\text{target}}$, which can only be evaluated after an initial training policy phase. Similarly, we do not train the policies beyond 100 epochs, as further training would cause overfitting to the retrieved data and yield non-meaningful scores.

Computational Overhead of Weight Estimation. Training each reference policy takes approximately 40 minutes, and all four are trained in parallel. Computing the relevance weights (Eq. 5) requires only 1.5 additional minutes, owing to the small size of the target dataset. In comparison, training the final policy takes about 2 hours. This overhead is modest, especially considering that it results in performance improvements on 7 out of 9 tasks (COLLAGE vs. NA-Fusion, Table 1).

A.4 Subsequence Dynamic Time Warping (S-DTW)

Dynamic Time Warping (DTW) is a classical algorithm for computing an optimal alignment between two sequences that may differ in length or exhibit temporal misalignment. Given sequences $X = \{x_1, \ldots, x_n\}$ and $Y = \{y_1, \ldots, y_m\}$, DTW computes a warping path $P = \{(i_k, j_k)\}_{k=1}^{L} \subseteq [1, n] \times [1, m]$ such that the cumulative cost of alignment is minimized:

$$\text{DTW}(X, Y) = \min_P \sum_{k=1}^{L} C(x_{i_k}, y_{j_k}),$$

where $C(x_i, y_j)$ is a local distance function between elements, typically the squared Euclidean distance between their embeddings. The path P must satisfy monotonicity and continuity constraints, ensuring a valid sequential alignment. The cumulative cost matrix $D \in \mathbb{R}^{n\times m}$ is computed recursively as:

$$D(i, j) = C(x_i, y_j) + \min\left\{D(i-1, j),\ D(i, j-1),\ D(i-1, j-1)\right\},$$

with suitable initialization at the borders.

Subsequence DTW (S-DTW) extends DTW to handle cases where one sequence (typically the query) is shorter than the other. Instead of aligning the full sequences, S-DTW searches for a contiguous subsequence of the longer sequence that best aligns with the entire shorter sequence. Formally, given

a shorter query sequence $X \in \mathbb{R}^{n \times d}$ and a longer reference sequence $Y \in \mathbb{R}^{m \times d}$, where $n < m$, the S-DTW distance is defined as:

$$\text{S-DTW}(X, Y) = \min_{1 \le s \le e \le m} \text{DTW}(X, Y_{s:e}),$$

where $Y_{s:e}$ denotes the subsequence of Y from index s to e. In practice, the cost is computed efficiently by evaluating DTW between X and all prefixes of Y, and taking the minimum over possible start and end points, as described in [43].

Application to Sub-Trajectory Retrieval: In imitation learning, particularly when learning from prior demonstrations, it is often desirable to match short sub-trajectories from a target task to relevant subsequences from longer demonstrations in a prior dataset. S-DTW provides a principled method to do so, as it allows aligning a fixed-length query trajectory to arbitrary contiguous segments in a reference trajectory.

In our work, we first segment each target demonstration into sub-trajectories using a velocity-based heuristic, as described in the main text. For each modality f, we use the associated encoder $\mathcal{F}_f$ to embed observations into a feature space. Given a target sub-trajectory $t' = \{o_1, \ldots, o_n\}$ and a longer prior trajectory $t = \{o'_1, \ldots, o'_m\}$, we compute a pairwise cost matrix $C \in \mathbb{R}^{n \times m}$, where:

$$C(i, j) = \left\| \mathcal{F}_f(o_i) - \mathcal{F}_f(o'_j) \right\|_2^2.$$

We then apply S-DTW to compute the minimal cumulative alignment cost between the entire query t' and any contiguous subsequence of t. This produces a similarity score between the target sub-trajectory and the reference. We repeat this process across the prior dataset and retrieve the top-K most similar subsequences per modality.

For efficient implementation and further details, we refer the reader to standard references on DTW and S-DTW [43] as well as the STRAP framework [15], which operationalizes this approach for sub-trajectory matching in the context of policy learning.