XSkill: Cross Embodiment Skill Discovery

Anonymous Author(s)
Affiliation
Address
email

Abstract: Human demonstration videos are a widely available data source for robot learning and an intuitive user interface for expressing desired behavior. However, directly extracting reusable robot manipulation skills from unstructured human videos is challenging due to the big embodiment difference and unobserved action parameters. To bridge this embodiment gap, this paper introduces XSkill, an imitation learning framework that 1) discovers a cross-embodiment representation called skill prototypes purely from unlabeled human and robot manipulation videos, 2) transfers the skill representation to robot actions using conditional diffusion policy, and finally, 3) composes the learned skill to accomplish unseen tasks specified by a human prompt video. Our experiments in simulation and real-world environments show that the discovered skill prototypes facilitate both skill transfer and composition for unseen tasks, resulting in a more general and scalable imitation learning framework. The performance of XSkill is best understood from the anonymous website: https://xskillcorl.github.io.

Keywords: Manipulation, Representation Learning, Cross-Embodiements

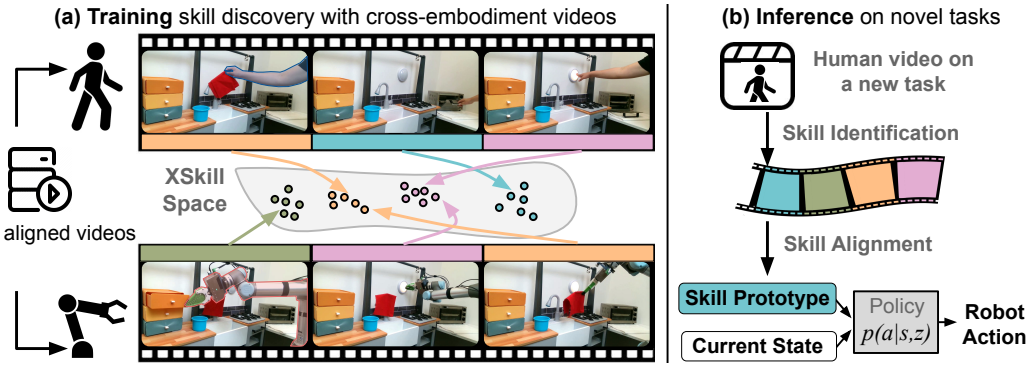


Figure 1: **Cross Embodiment Skill Discovery.** XSkill first learns a cross-embodiment skill representation space (XSkill Space on the left). During inference, given a human demonstration of unseen tasks, XSkill first identifies the human skills by projecting the video demonstration onto the learned cross-embodiment skill representation space. The identified skills are then executed by the skill-conditioned visuomotor policy.

1 Introduction

A successful imitation learning algorithm from human demonstration is enabled by three critical capabilities: 1) **Discover**, decomposing the demonstrated task into a set of sub-tasks and identifying the common and reusable skills required to accomplish these sub-tasks. 2) **Transfer**, mapping each of the observed skills to its own embodiment, which is different from that of the demonstrator. 3) **Compose**, performing novel compositions of the learned skills to accomplish new tasks.

This paper addresses these critical capabilities by decomposing and identifying appropriate "skills" from human demonstration so that they are *transferable* to robots and *composable* to perform new

Submitted to the 7th Conference on Robot Learning (CoRL 2023). Do not distribute.

tasks. We refer to the task as **"Cross-Embodiment Skill Discovery"** and introduce our method **XSkill** for this task. At its core, XSkill learns a shared embedding space for the robot and human skills through self-supervised learning [1, 2, 3, 4, 5]. The algorithm extracts features for *unaligned* human and robot video sequences, where video clips share similar action effects (i.e., similar skill) should result in a closer feature distance. To encourage across-embodiment alignment, we introduce a set of learnable skill prototypes through feature clustering. These prototypes act as representative anchors in the continuous embedding space. By force to share the same set of prototypes, we could effectively align the skill representations between embodiments.

With the identified cross-embodiment skill prototypes, the robot can then learn a skill-conditioned visuomotor policy that transfers each identified skill to the robot's action space. During inference, the algorithm can one-shot generalize to new tasks using the learned skills, where the new task is defined by a single human demonstration (i.e., prompt video). With the proposed skill alignment transformer, the algorithm can robustly align skills in the human video to the robot visual observation, despite the embodiment difference and unexpected execution failures.

Our approach improves upon the direct imitation learning method [6] by decomposing the complex long-horizon tasks into a set of reusable skills (i.e., low-level visuomotor policies), which is much easier to learn and generalizable to new tasks through composition. Meanwhile, our approach differs from existing work on single-embodiment skill discovery [7, 8, 9], which solely relies on on-robot demonstration data. By learning cross-embodiment skill prototypes, our framework can use direct human demonstration, which is more cost-effective and scalable, even for non-expert demonstrators.

In summary, our contributions are as follows:

- We formulate the task of **cross-embodiment skill discovery**, a useful and essential building block for imitation learning. Together with the new cross-embodiment dataset in simulation and real world, we hope to inspire future exploration in this area.
- Introducing the first attempt toward this task **XSkill** that consists of three novel components: 1) A self-supervised representation learning algorithm that discovers a set of share skill prototypes from the unlabeled robot and human videos. 2) A skill-conditioned diffusion policy that translates the observed human demonstration into robot actions. 3) A skill composition framework (with skill alignment transformer) to robustly detect, align and compose the learned skills to accomplish new tasks from a single human demonstration.

Our experiments in simulation and real-world environments show that the discovered skill prototypes facilitate both skill transfer and composition for unseen tasks, resulting in a more general and scalable imitation learning framework. The dataset and code will be publicly available.

2 Related Work

Robot Skill Discovery. A large body of works have been proposed for discover robot skill through via option frameworks [10, 7, 11, 12, 13, 14] or through the lens of mutual information [15, 8, 16, 17, 18, 19, 20]. Most of these works requires interacting with environment and yield high sample complexity. To ease the sample complexity, the other line of works [9, 21, 22, 23] have explored discover skill directly from robot demonstration data. The majority of those prior works require physical state or robot action trajectories. BUDS [24] eases this requirement by discovering skills through raw RGB data. Unlike these prior works which discover skills only for a single embodiment, XSkill explores skill discovery in a cross-embodiment setting.

Imitation learning. Learning robot behavior from demonstration data is a longstanding challenges [6]. Prior works on imitation learning has shown promising results on realworld robot manipulation task through explicit policy [25, 26, 27, 28, 29, 30, 31, 32], implicit policy [33] or diffusion model [34, 35, 36]. Our works utilize the hierarchical imitation learning framework [37, 38, 39, 40, 41] to transfer the discovered skills through skill-conditioned diffusion policy[34]. XSkill can be categorized as one-shot imitation learning [42]. The most of prior works imitate from the same-embodiment demonstration [43] or from a different embodiment [44, 45] but require task label

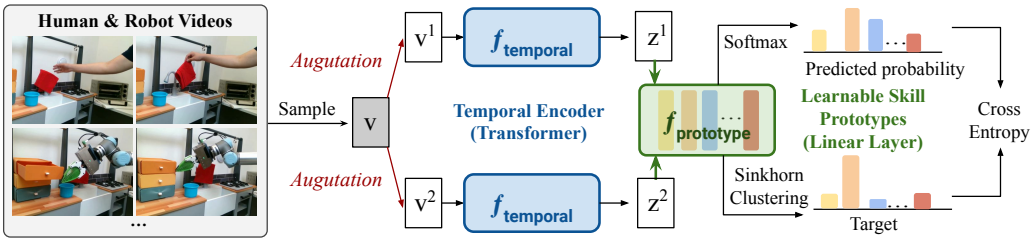


Figure 2: **XSkill Discover:** At each training iteration, a batch of video are sampled from the same embodiment dataset. Each video v_i^t is augmented into two versions and encoded using temporal encoder f_{temporal}. The learnable skill prototypes $f_{\text{prototype}}$ are implemented as a normalized linear layer without bias. Both f_{temporal} and $f_{\text{prototype}}$ are trained jointly to minimize the CorssEntorpy loss between the predicted and target the probability of skill prototypes. Sinkhorn regularization is applied to the target probability, ensuring all prototypes are used for each batch (same embodiment), thereby encouraging prototype sharing across embodiments.

during the training. In contrast, XSkill does not require any task label or correspondence between embodiments during training to accomplish one-shot imitation from human demonstration.

Learning from human video. A number of works [46, 47, 48, 49, 50] have studied leveraging human videos to learn robotic policy. One approach is to construct reward functions from human video through domain translation [47, 51, 52, 53, 54, 55], video classifier [49, 56], or state representation learning [57, 48, 58]. Despite showing promising results on learning from cross-embodiment demonstration, most of these works require reinforcement learning (RL) to learn policy based on the constructed reward functions, which is expensive to deploy in real world. In contrast to the majority of these works, our method does not involve RL in the loop and focus on one-shot imitation from human videos. Bahl et al. [59] proposed to initialize policy through human prior but still requires interaction with environment to improve the policy. Our work is related to Yu et al. [50, 60], which explored one-shot imitation through meta-learning. Unlike meta-learning approach, our method does not require any task pairing information during the training. More recently, MimicPlay [61] proposed leveraging human videos to learn a cross-embodiment plan latent space. MimicPlay can only complete unseen tasks through robot prompt video. In contrast, XSkill learns a cross-embodiment skill space and policy execution only requires skill abstraction. As a result, XSkill can achieve one-shot learning through human prompt video.

3 Approach

The XSkill framework consists of three phases: Discover §3.1, Transfer §3.2, and Compose §3.3 that uses three different data sources. In the discover phase, the algorithm has access to a **human demonstration dataset** $\mathcal{D}^h$ and a **robot teleoperation dataset** $\mathcal{D}^r$ to discover a cross-embodiment skill representation space $\mathcal{Z}$. This space, along with K common learnable skill prototypes, is learned through self-supervised learning. Both datasets are **unsegmented** and **unaligned** and each video in the dataset performs a subset of N skills. In the transfer phase, the algorithm uses the robot teleoperation dataset $\mathcal{D}^r$ to learn the skill-conditioned visuomotor policy $P(a|s, z)$, where $z \in \mathcal{Z}$ and s includes both robot proprioception and visual observation o. In the Compose phase, the algorithm takes as input a single **human prompt video** $\tau^h_{\textbf{prompt}}$ for a new task that requires an unseen composition of skills to complete. From this video prompt, the algorithm first identifies the order of skills used in the prompt and then composes the skills using the learned policy $P(a|s, z)$.

3.1 Discover: Learning Shared Skill Prototypes

As the first step, XSkill aims to discover skills in a self-supervised manner such that the learned visual representation of the **same skills** executed by **different embodiments** can be close in the cross-embodiment skill representation space $\mathcal{Z}$. Off-the-shelf vision representations are often insufficient since they are often sensitive to the visual appearance of the agent or environment. Instead, we want the learned skill representation to focus on the underlying skills being performed. To achieve this goal, XSkill introduces two key ideas:

- Learning a set of shared skill prototypes through clustering. These prototypes act as representative anchors in the continuous embedding space. Then by force to share the same set of prototypes, we could effectively align the skill representations between embodiment.

- Regularizing the training process using Sinkhorn-Knopp clustering [62, 1] within single-embodiment batches. Together, they ensure all prototypes are used for each batch (all from the same embodiment). This design avoids the degeneration case where different embodiment maps to different prototypes, thereby ensuring prototype sharing.

Skill representations. XSkill extracts skill representations z using **human and robot videos** from $\mathcal{D}^h$ and $\mathcal{D}^r$, mapping them into a shared representation space $\mathcal{Z}$. To mitigate variations in execution speed across different embodiments, we sample M frames uniformly from each video V_i and construct video clips $\{v_{ij}\}_{j=0}^{M}$ using a moving window of length L. Then, we extract the skill representation $z_{ij} = f_{\text{temporal}}(v_{ij})$ from each video clip with a temporal skill encoder consisting of a vision backbone and a transformer encoder [63]. We append a learnable representation token [64, 65] into the sequence to better capture the motion across frames.

Skills as Prototypes. Once the skill representations are obtained from demonstration videos, XSkill maps representations from all embodiments to a set of K **skill prototypes** $\{c_k\}_{k=1}^{K}$, where each is a learnable vector. The skill prototypes are implemented as a normalized linear layer $f_{\text{prototype}}$ without bias. This mapping is accomplished through a self-supervised learning framework similar to SwAV [1]. The XSkill starts by augmenting the video clip v_{ij} using a randomly selected transformation before feeding into f_{temporal} (e.g., random crop). Subsequently, XSkill projects the normalized representation $\frac{z_{ij}}{||z_{ij}||_2}$ onto the set of learnable skill prototypes $\{c_k\}_{k=1}^{K}$. The probability p_{ij} of skills being executed in the given video clip v_{ij} is predicted by applying the Softmax function. The target distribution q_{ij} is obtained from the other augmented version of the same video clip. The target probability, instead of applying the Softmax function over projection, is obtained by running online clustering Sinkhorn-Knopp algorithm, which we describe later in the paper. Both f_{temporal} and $f_{\text{prototype}}$ are trained jointly to minimize the CorssEntorpy loss between the predicted p_{ij} and target q_{ij} skill prototypes distributions: $\mathcal{L}_{\text{prototype}} = -\sum_{i=1}^{B}\sum_{j=0}^{M}\sum_{k=1}^{K} q_{ij}^{(k)} \log p_{ij}^{(k)}$, where B is batch size.

Learning Aligned Skill Prototypes. To ensure that the prototypes focus on the underlying skills rather than embodiment and environment appear, XSkill combines data sampling and entropy regularization in the training process. At each training iteration, XSkill samples video clips from the same embodiment and constructs a batch. This batch is then passed into the framework shown in Fig. 2. By clustering the feature from the same embodiment, the embodiment difference is disregarded.

To achieve alignment of prototypes across different embodiments, we compute the target probability distribution q_{ij} by incorporating the Sinkhorn-Knopp clustering algorithm [62]. This algorithm ensures that the skill representations within a batch are evenly assigned to each skill prototype. The input to the Sinkhorn-Knopp algorithm is the normalized skill projection matrix $C^T Z$, which is iteratively modified to produce a double stochastic matrix where the columns are set as the target probabilities q_{ij}. Intuitively, the Sinkhorn-Knopp clustering can be viewed as an entropy regularization that encourages the model to utilize as many skill prototypes as possible for the same embodiment, thereby promoting prototype alignment across embodiments.

Time Contrastive Learning. XSkill utilizes a time contrastive loss [66, 67] in order to encapsulate the temporal structure of skills within video demonstrations. It posits that skill prototype probabilities should be similar for video clips closer in time and dissimilar for those farther apart. Specifically, This is achieved by establishing a positive window w_p and a negative window w_n. For a given clip v_{ix} at time x in video V_i, a positive sample v_{iy} is chosen within w_p from time x, and a negative sample v_{iz} is selected outside w_n. XSkill minimizes the following InfoNCE loss [68]: $\mathcal{L}_{tcn} = -\sum_{i=1}^{B} \log \frac{\exp(\mathcal{S}(p_{ix}, p_{iy})/\tau_{\text{tcn}})}{\exp(\mathcal{S}(p_{ix}, p_{iy})/\tau_{\text{tcn}}) + \exp(\mathcal{S}(p_{ix}, p_{iz})/\tau_{\text{tcn}})}$, wherein $\mathcal{S}$ is the measure of similarity, which is implemented as dot product in XSkill and τ_{tcn} is the temperature. Here, p_{ix}, p_{iy}, p_{iz} represent the skill prototype probabilities for clips v_{ix}, v_{iy}, and v_{iz}, respectively.

3.2 Transfer: Skill conditioned imitation learning

To transfer skill representations into concrete robot actions, we train a skill-conditioned visuomotor policy using imitation learning. More specifically, our approach builds upon diffusion policy [34]

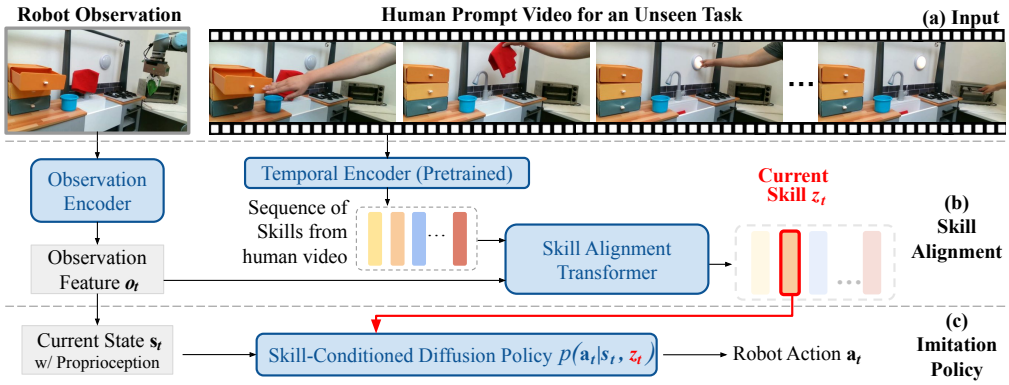


Figure 3: **Transfer & Composition:** During inference, a human demonstration of a new task is given, XSkill first extracts a sequence of skills, which can be viewed as a high-level task plan. However, this plan is not immediately aligned with robot execution speed due to the embodiment gap. Therefore we need to align the plan based on the robot's current observation, which is achieved by the Skill Alignment Transformer. The inferred skills are then passed into a skill-conditioned diffusion policy to get the robot's actions.

which uses Denoising Diffusion Probabilistic Models (DDPMs) [69] to represent the multimodal action distributions from human teleoperation demonstration. Diffusion policy has been shown to be stable to train and work well with a small amount of data, both are essential for our tasks.

Our diffusion-based imitation learning policy $\pi(\mathbf{a_t}|s_t, z_t)$ is trained with robot teleoperation dataset $\mathcal{D}^r$, where $\mathbf{a_t}$ denotes an action-sequence $\{a_t, ..., a_{t+L}\}$ of length L starting from state s_t. The diffusion policy takes skill representation z_t and state s_t which includes robot proprioception and visual observation o_t as input and produces an action-sequence $\mathbf{a_t}$. The skill representation z_t is computed with trained f_{temporal} using $v_t = \{o_t, o_{t+1}..., o_{t+L}\}$ as we described in §3.1.

3.3 Compose: Performing unseen task from one-shot human prompt video

Once skills have been discovered and transferred into robot manipulations through imitation learning, our object is to compose the skills to solve unseen tasks based on a human prompt video which contains a demonstration by a human on how to complete an unseen task. To do so, XSkill first maps a human prompt video with length T_{prompt} into the cross-embodiment skill representation space $\mathcal{Z}$ using learned f_{temporal}. This generates a sequence of skill representation for the demonstrated task, denoted as $\tilde{z} = \{z_t\}_{t=0}^{T_{\text{prompt}}}$, which is essentially a task execution plan. The robot can complete the task by sequentially executing the skills in the plan by querying the skill-conditioned diffusion policy.

However, directly following the skill sequence $\tilde{z}$ for execution often results in a fragile system that is sensitive to unexpected failures or speed mismatch. For example, if it fails to open a light, it needs to retry the skill to succeed. If the robot sequentially executes the $\tilde{z}$, it will proceed to execute the next skill without correcting the error. Therefore to improve the system's robustness, we introduce a Skill Alignment Transformer (SAT) that is described below.

Skill Alignment Transformer (SAT). The Skill Alignment Transformer, denoted as $\phi(z_t|o_t, \tilde{z})$, aligns the robot with the intended skill execution based on its current state. The key idea is that by analyzing the current state and comparing it to the skill sequence $\tilde{z}$, the robot can determine which part of the skills has already been executed and infer the most likely skill to be executed next. This alignment process allows the robot to synchronize its task execution with the demonstrated task in the human prompt, thereby minimizing discrepancies arising from variations in execution speed between robots and humans and offering robustness against execution failures.

As illustrated in Fig. 3, SAT regards each skill in the skill sequence $\tilde{z}$ as a skill token. It employs a state encoder, $f_{\text{state-encoder}}$, to convert the robot's current visual observation into a state token. The skill and state tokens with position encoding, are then passed into a transformer encoder to predict the skill that needs to be executed next. Within the transformer, the state token can attend to each skill token to determine whether the skill has been executed given the current state information. For example, if the light is already open in the current state, the skill to open the light should not be considered the next skill to be executed.

To train SAT $\phi(z_t|o_t, \tilde{z})$, we sample a full trajectory from the robot teleoperation dataset and extract the skill sequence $\tilde{z}$. This is achieved by passing robot trajectory video clips $\{v_t\}_{t=0}^{T}$, where $v_t = \{o_t, o_{t+1}..., o_{t+L}\}$, through the temporal skill encoder, yielding $\{z_t\}_{t=0}^{T}$. The length of the sampled trajectory is represented by T. Next, a time index t is chosen randomly within the range $[0, T]$. Our system predicts $\hat{z}_t$ using the skill sequence $\tilde{z}$ and visual observation o_t as inputs to SAT. We optimize the model by minimizing the mean square error between $\hat{z}_t$ and the actual z_t.

4 Evaluation

Environment. We test XSkill on both simulated and real-world environments:

- **Franka Kitchen:** is a simulated kitchen environment [70] that includes 7 sub-tasks and is accompanied by 580 robot demonstration trajectories. To create a cross-embodiment demonstration, we construct a sphere agent that is visually significantly different from the original robot. To further increase the domain gap, we sub-sample sphere agent demonstrations to emulate the execution speed differences. During the inference, the robot must complete an unseen composition of sub-tasks after viewing a prompt video from the sphere agent demonstration.
- **Realworld Kitchen:** is a new benchmark we introduce to evaluate algorithm performance on physical robot hardware. The dataset contains four sub-tasks, namely, opening the oven, grasping a cloth, closing a drawer, and turning on a light. We have recorded 175 human demonstrations and 175 teleoperation demonstrations. Each demonstration completes three sub-tasks in a randomly determined order. During inference, the robot is required to complete an unseen composition of either three or four sub-tasks after observing a prompt video taken from a human demonstration.

Baselines. We compare XSkill with the following baselines:

- **GCD Policy:** Instead of using skill-conditioned policy, we compare to a goal-conditioned diffusion policy $\pi(\mathbf{a_t}|s_t, g_t)$, where the goal image g_t is the image in prompt video H steps after the current time t after alignment. The alignment is done using nearest-neighbor matching between robot observation and prompt video in the embedding space, where the embedding is trained jointly with the policy.
- **GCD Policy w. TCN:** Same as the GCD Policy above but replacing the video encoder with pre-trained Time-Contrastive Network (TCN)[66].
- **XSkill w. NN-composition:** XSkill removing skill aligment transformer. Instead, find the alignment using the nearest neighbor image between the robot observation and the prompt video. The image embedding is extracted using the same encoder f_{temporal} as XSkill.
- **XSkill w.o proto. loss:** XSkill removing prototype loss $\mathcal{L}_{\text{prototype}}$ in the discover phase.

Implementation Details. We set the number of prototypes K as 128 in the simulated environment and 32 prototypes in the real-world environment. The video clip length L and uniform sample frames M are set as 8 and 100 for both simulated and real-world kitchens. The ablation study on K and more implementation details can be found in the supplementary material.

Evaluation protocol. During inference, the robot is required to accomplish the sub-tasks in the *same order* as demonstrated in the prompt video. The performance of XSkill and all baseline methods is evaluated based on both sub-task completion and order of completion. If the robot executes an undemonstrated sub-task, the episode ends. The evaluation metric is denoted as $\frac{\text{number of sub-tasks completed}}{\text{number of total sub-tasks}}$. In the simulation, each method is trained using three distinct seeds and tested under 32 unique initial environment conditions during the inference. In the real-world kitchen, each task is assessed 10 times under varying initial environment conditions during the inference.

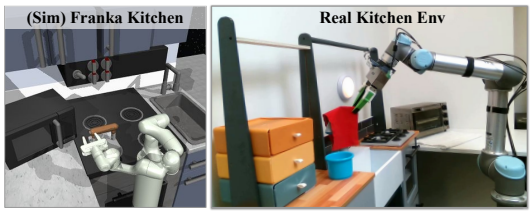
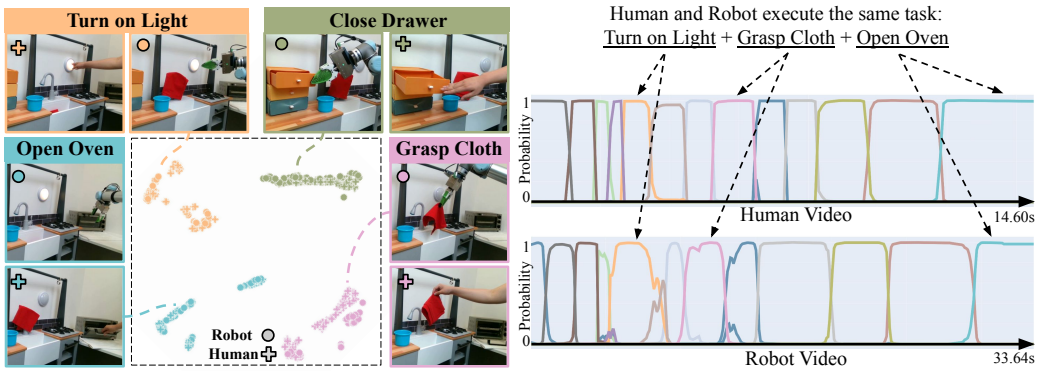


Figure 4: **Evaluation Environments.**

4.1 Key Findings

XSkill can learn cross-embodiment skill representation. [XSkill] learns cross-embodiment skill representation by effectively extracting reusable skills from demonstrations including both in-



(a) t-SNE Visualization of Skill Space **(b) Skill Prototype Distribution Over Time**

Figure 5: **XSkill embedding. (a)** We utilize t-SNE visualization to showcase the alignment of skill representations among various embodiments when in contact with the same object. **(b)** We present projected prototypes for both human and robot executing identical tasks. XSkill achieves efficient alignment of representations, not just during physical contact, but also during the transition between manipulating different objects.

contact manipulation with various objects and also the transitions between them. In Fig. 5(a), we visualize these learned skill representations using t-SNE, where the skills executed by the human and robot to manipulate the same object are grouped together and clearly separated from the others. Additionally, we visualize the projected prototypes for human and robot completion of the same task in Fig. 5(b). Despite differences in execution speed (humans execute $\times$ 2 faster), [XSkill] can decompose the task into meaningful and aligned skill prototypes for both in-contact manipulations and transitions between them. Consequently, the performance of [XSkill] with cross-embodiment prompts only drops around 5%, compared to using the same embodiment prompt (Tab. 2 seen tasks).

XSkill can generalize the imitation policy to unseen tasks. [XSkill] achieves 70.2% and 60% success (Tab. 1 & 2) on unseen tasks with cross-embodiment prompts in simulated and real-world environments, which outperforms all baselines. As shown in Fig. 6 (a), [XSkill] is capable of decomposing unseen tasks into sequences of previously seen skill abstractions which can executable by the learned skill-conditional imitation policy. As a result, [XSkill] enables one-shot imitation learning through skill decomposition and re-composition. The performance of [XSkill] slightly drops in the real world due to novel transition dynamics presented in prompts and a scarcity of collected robot data. For instance, the robot struggles to complete tasks involving grasping the cloth followed by closing the drawer, since no such transition dynamics are present in the collected robot teleoperation dataset. In summary, [XSkill] demonstrates promising task generalization, but its efficacy is still limited by the diversity in the robot teleoperation data.

Table 1: **Simulation Result (%)**

	Same	Cross Embodiment		
Execution speed	$\times$ 1	$\times$ 1	$\times$ 1.3	$\times$ 1.5
GCD Policy	91.4	0.00	0.00	0.00
GCD Policy w. TCN	2.50	3.55	2.00	1.25
XSkill w. NN-compose	93.7	61.2	23.4	15.2
XSkill w.o proto. loss	80.1	56.3	12.5	3.75
XSkill	**95.8**	**89.4**	**83.7**	**70.2**

Skill prototypes are essential for cross-embodiment learning. To assess the importance of shared skill prototypes, we compare our approach with [XSkill w.o proto. loss] in simulation. [XSkill] outperforms this baseline significantly with cross-embodiment prompts. Further, we observe that the performance of [XSkill w.o proto. loss] deteriorates rapidly when the cross-embodiment agent operates at a higher speed. These results suggest that skill prototypes not only facilitate the learning of morphology-invariant representations but also avoid learning a speed-sensitive one.

Table 2: **Real-world Result (%)**

	3 Subtasks				4 Subtasks
	Same		Cross		Cross
	Seen	Unseen	Seen	Unseen	Unseen
GCD policy	68.3	53.3	0.00	0.00	0.00
GCD w. TCN	25.0	22.2	26.7	23.3	15.6
XSkill	**86.7**	**80.0**	**81.7**	**76.7**	**60.0**

SAT can align skill based on task progress. [XSkill] outperforms [XSkill w. NN-composition] by more than 50% with cross-embodiment prompt. The performance of NN-composition significantly declines when the cross-embodiment agent executes at a much faster speed (Tab. 1). We down-

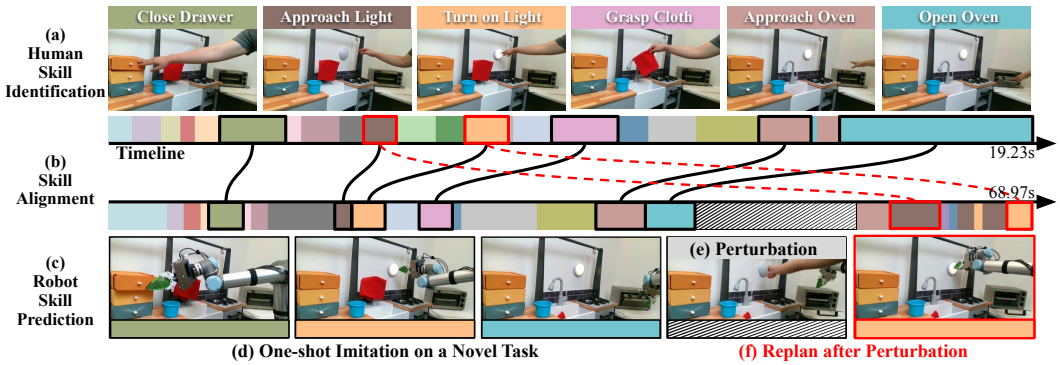


Figure 6: **Execution on a novel task and robustness to perturbation.** **(a)** XSkill analyzes a human video of a novel task, identifying skills for each timestep (represented by distinct colors). **(c)** The robot leverages this analysis to predict the appropriate skill based on the current observation and subsequently executes the corresponding skill. Skill alignment **(b)** is critical to handle execution speed difference caused by cross embodiment. **(e)** With appropriate skill conditions at each step, the robot achieves one-shot imitation of the novel task. **(e)** Deliberately introducing a perturbation, a human manually turns off the light. **(f)** The robot accurately predicts the necessary skills and adaptively replans the execution to successfully reach the goal state once again. Please check out the anonymous website for videos: `https://xskillcorl.github.io`

sample cross-embodiment demonstrations and also prompt videos with different ratio. For instance, a ratio of $\times 1.5$ emulates that human execute $\times 1.5$ faster than the robot in the real world. This can be attributed to two main factors. Firstly, relying solely on visual representation is prone to distraction due to morphological differences. Second, when the demonstration speed in the prompt videos is much faster than the robot's execution speed, certain states might not be captured in the prompt video. As a result, the retrieved skills can become inaccurate and unstable. In contrast, as illustrated in Fig. 6, the SAT enhances the robustness of [XSkill] to the demonstration speed and enables adaptive adjustment of skills based on the current state of the robot and the task progress.

Sequential input benefits cross-embodiment learning. Unlike [GCD Policy w. TCN], which relies on TCN encoding for single images, [XSkill] utilizes sequential image input to generate representations. In the cross-embodiment scenario, [GCD Policy w. TCN] outperforms vanilla [GCD Policy], indicating partial bridging of the embodiment gap through TCN embedding. However, [GCD Policy w. TCN] cannot follow the sub-tasks completion order in the prompt video, resulting in a lower evaluation score. This suggests that TCN embedding does not completely align the representations.

4.2 Limitation and Future Work

One limitation of XSkill is the requirement of specifying the number of prototypes as an input to the algorithm. While our ablation study demonstrates that XSkill is not highly sensitive to this number, fine-tuning the hyperparameters may still be necessary to achieve optimal performance based on the specific dataset and its usage. In addition, while XSkill does not require labeled correspondence between human and robot datasets, our current benchmark only contains videos recorded using the same camera under the same laboratory environment. Future works may explore the feasibility of our method under a wider variety of camera setup and environments, taking advantage of the widely available YouTube videos and multi-environment datasets [71].

5 Conclusion

We introduce XSkill for the task of cross-embodiment skill discovery. This framework extracts common manipulation skills from unstructured human, robot videos in a way that is transferrable to robots and composable to perform new tasks. Extensive experiments in simulation and real-world demonstrate that XSkill improves upon the direct behavior cloning method especially complex long-horizon tasks. Moreover, by leveraging cross-embodiment skill prototypes, XSkill can directly leverage non-expert human demonstration for new tasks definition, making the framework much more cost-effective and scalable.

References

[1] M. Caron, I. Misra, J. Mairal, P. Goyal, P. Bojanowski, and A. Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *Advances in neural information processing systems*, 33:9912–9924, 2020.

[2] Y. M. Asano, C. Rupprecht, and A. Vedaldi. Self-labelling via simultaneous clustering and representation learning. *arXiv preprint arXiv:1911.05371*, 2019.

[3] M. A. Bautista, A. Sanakoyeu, E. Tikhoncheva, and B. Ommer. Cliquecnn: Deep unsupervised exemplar learning. *Advances in Neural Information Processing Systems*, 29, 2016.

[4] M. Caron, P. Bojanowski, A. Joulin, and M. Douze. Deep clustering for unsupervised learning of visual features. In *Proceedings of the European conference on computer vision (ECCV)*, pages 132–149, 2018.

[5] D. Yarats, R. Fergus, A. Lazaric, and L. Pinto. Reinforcement learning with prototypical representations. 2021.

[6] B. Argall, S. Chernova, M. M. Veloso, and B. Browning. A survey of robot learning from demonstration. *Robotics Auton. Syst.*, 57:469–483, 2009.

[7] A. Bagaria and G. Konidaris. Option discovery using deep skill chaining. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=B1gqipNYwH.

[8] B. Eysenbach, A. Gupta, J. Ibarz, and S. Levine. Diversity is all you need: Learning skills without a reward function. *ArXiv*, abs/1802.06070, 2018.

[9] A. Sharma, S. S. Gu, S. Levine, V. Kumar, and K. Hausman. Dynamics-aware unsupervised discovery of skills. *ArXiv*, abs/1907.01657, 2019.

[10] R. S. Sutton, D. Precup, and S. Singh. Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. *Artif. Intell.*, 112:181–211, 1999.

[11] R. Fox, S. Krishnan, I. Stoica, and K. Goldberg. Multi-level discovery of deep options. *ArXiv*, abs/1703.08294, 2017.

[12] G. D. Konidaris and A. G. Barto. Skill discovery in continuous reinforcement learning domains using skill chaining. In *NIPS*, 2009.

[13] V. C. V. Kumar, S. Ha, and C. K. Liu. Expanding motor skills using relay networks. In *Conference on Robot Learning*, 2018.

[14] J. Achiam, H. Edwards, D. Amodei, and P. Abbeel. Variational option discovery algorithms, 2018.

[15] K. Gregor, D. J. Rezende, and D. Wierstra. Variational intrinsic control. *ArXiv*, abs/1611.07507, 2016.

[16] K. Hausman, J. T. Springenberg, Z. Wang, N. M. O. Heess, and M. A. Riedmiller. Learning an embedding space for transferable robot skills. In *International Conference on Learning Representations*, 2018.

[17] T. Shankar and A. Gupta. Learning robot skills with temporal variational inference. In *Proceedings of (ICML) International Conference on Machine Learning*, pages 8624 – 8633, July 2020.

[18] L. Lee, B. Eysenbach, E. Parisotto, E. Xing, S. Levine, and R. Salakhutdinov. Efficient exploration via state marginal matching, 2020. URL https://openreview.net/forum?id=Hkla1eHFvS.

[19] H. Liu and P. Abbeel. Aps: Active pretraining with successor features. In *International Conference on Machine Learning*, 2021.

[20] M. Laskin, H. Liu, X. B. Peng, D. Yarats, A. Rajeswaran, and P. Abbeel. Unsupervised reinforcement learning with contrastive intrinsic control. 35:34478–34491, 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/debf482a7dbdc401f9052dbe15702837-Paper-Conference.pdf.

[21] D. Tanneberg, K. Ploeger, E. Rueckert, and J. Peters. SKID RAW: skill discovery from raw trajectories. *IEEE Robotics and Automation Letters*, 6(3):4696–4703, 2021. doi:10.1109/LRA.2021.3068891.

[22] A. Singh, H. Liu, G. Zhou, A. Yu, N. Rhinehart, and S. Levine. Parrot: Data-driven behavioral priors for reinforcement learning. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=Ysuv-WOFeKR.

[23] K. Pertsch, Y. Lee, and J. J. Lim. Accelerating reinforcement learning with learned skill priors. In *Conference on Robot Learning (CoRL)*, 2020.

[24] Y. Zhu, P. Stone, and Y. Zhu. Bottom-up skill discovery from unsegmented demonstrations for long-horizon robot manipulation. *IEEE Robotics and Automation Letters*, 7:4126–4133, 2021.

[25] A. Mandlekar, D. Xu, J. Wong, S. Nasiriany, C. Wang, R. Kulkarni, L. Fei-Fei, S. Savarese, Y. Zhu, and R. Mart'in-Mart'in. What matters in learning from offline human demonstrations for robot manipulation. In *Conference on Robot Learning*, 2021.

[26] P. R. Florence, L. Manuelli, and R. Tedrake. Self-supervised correspondence in visuomotor policy learning. *IEEE Robotics and Automation Letters*, 5:492–499, 2019.

[27] R. Rahmatizadeh, P. Abolghasemi, L. Bölöni, and S. Levine. Vision-based multi-task manipulation for inexpensive robots using end-to-end learning from demonstration. *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3758–3765, 2017.

[28] A. Zeng, P. R. Florence, J. Tompson, S. Welker, J. Chien, M. Attarian, T. Armstrong, I. Krasin, D. Duong, V. Sindhwani, and J. Lee. Transporter networks: Rearranging the visual world for robotic manipulation. In *Conference on Robot Learning*, 2020.

[29] T. Zhang, Z. McCarthy, O. Jow, D. Lee, K. Goldberg, and P. Abbeel. Deep imitation learning for complex manipulation tasks from virtual reality teleoperation. *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1–8, 2017.

[30] N. M. M. Shafiullah, Z. J. Cui, A. Altanzaya, and L. Pinto. Behavior transformers: Cloning k modes with one stone. In *Thirty-Sixth Conference on Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=agTr-vRQsa.

[31] Y. Zhu, A. Joshi, P. Stone, and Y. Zhu. VIOLA: Object-centric imitation learning for vision-based robot manipulation. In *6th Annual Conference on Robot Learning*, 2022. URL https://openreview.net/forum?id=L8hCfhPbFho.

[32] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, J. Dabis, C. Finn, K. Gopalakrishnan, K. Hausman, A. Herzog, J. Hsu, J. Ibarz, B. Ichter, A. Irpan, T. Jackson, S. Jesmonth, N. Joshi, R. Julian, D. Kalashnikov, Y. Kuang, I. Leal, K.-H. Lee, S. Levine, Y. Lu, U. Malla, D. Manjunath, I. Mordatch, O. Nachum, C. Parada, J. Peralta, E. Perez, K. Pertsch, J. Quiambao, K. Rao, M. Ryoo, G. Salazar, P. Sanketi, K. Sayed, J. Singh, S. Sontakke, A. Stone, C. Tan, H. Tran, V. Vanhoucke, S. Vega, Q. Vuong, F. Xia, T. Xiao, P. Xu, S. Xu, T. Yu, and B. Zitkovich. Rt-1: Robotics transformer for real-world control at scale. In *arXiv preprint arXiv:2204.01691*, 2022.

[33] P. Florence, C. Lynch, A. Zeng, O. Ramirez, A. Wahid, L. Downs, A. Wong, J. Lee, I. Mordatch, and J. Tompson. Implicit behavioral cloning. *Conference on Robot Learning (CoRL)*, 2021.

[34] C. Chi, S. Feng, Y. Du, Z. Xu, E. Cousineau, B. Burchfiel, and S. Song. Diffusion policy: Visuomotor policy learning via action diffusion. *arXiv preprint arXiv:2303.04137*, 2023.

[35] T. Pearce, T. Rashid, A. Kanervisto, D. Bignell, M. Sun, R. Georgescu, S. V. Macua, S. Z. Tan, I. Momennejad, K. Hofmann, and S. Devlin. Imitating human behaviour with diffusion models. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=Pv1GPQzRrC8.

[36] M. Reuss, M. X. Li, X. Jia, and R. Lioutikov. Goal-conditioned imitation learning using score-based diffusion policies. *ArXiv*, abs/2304.02532, 2023.

[37] C. Lynch, M. Khansari, T. Xiao, V. Kumar, J. Tompson, S. Levine, and P. Sermanet. Learning latent plans from play. In *Conference on Robot Learning*, 2019.

[38] H. M. Le, N. Jiang, A. Agarwal, M. Dudík, Y. Yue, and H. D. I. au2. Hierarchical imitation and reinforcement learning, 2018.

[39] A. Mandlekar, F. Ramos, B. Boots, L. Fei-Fei, A. Garg, and D. Fox. Iris: Implicit reinforcement without interaction at scale for learning control from offline robot manipulation data. *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4414–4420, 2019.

[40] A. Mandlekar, D. Xu, R. Martín-Martín, S. Savarese, and L. Fei-Fei. Learning to generalize across long-horizon tasks from human demonstrations. *ArXiv*, abs/2003.06085, 2020.

[41] K. Shiarlis, M. Wulfmeier, S. Salter, S. Whiteson, and I. Posner. Taco: Learning task decomposition via temporal alignment for control. In *International Conference on Machine Learning*, 2018.

[42] Y. Duan, M. Andrychowicz, B. C. Stadie, J. Ho, J. Schneider, I. Sutskever, P. Abbeel, and W. Zaremba. One-shot imitation learning. In *NIPS*, 2017.

[43] Z. Mandi, F. Liu, K. Lee, and P. Abbeel. Towards more generalizable one-shot visual imitation learning. *2022 International Conference on Robotics and Automation (ICRA)*, pages 2434–2444, 2021.

[44] C. Finn, T. Yu, T. Zhang, P. Abbeel, and S. Levine. One-shot visual imitation learning via meta-learning. *ArXiv*, abs/1709.04905, 2017.

[45] S. Dasari and A. K. Gupta. Transformers for one-shot visual imitation. *ArXiv*, abs/2011.05970, 2020.

[46] T. Xiao, I. Radosavovic, T. Darrell, and J. Malik. Masked visual pre-training for motor control. *ArXiv*, abs/2203.06173, 2022.

[47] Y. Liu, A. Gupta, P. Abbeel, and S. Levine. Imitation from observation: Learning to imitate behaviors from raw video via context translation, 2018.

[48] K. Zakka, A. Zeng, P. R. Florence, J. Tompson, J. Bohg, and D. Dwibedi. Xirl: Cross-embodiment inverse reinforcement learning. In *Conference on Robot Learning*, 2021.

[49] L. Shao, T. Migimatsu, Q. Zhang, K. Yang, and J. Bohg. Concept2robot: Learning manipulation concepts from instructions and human demonstrations. *The International Journal of Robotics Research*, 40(12-14):1419–1434, 2021.

[50] T. Yu, C. Finn, A. Xie, S. Dasari, T. Zhang, P. Abbeel, and S. Levine. One-shot imitation from observing humans via domain-adaptive meta-learning, 2018.

[51] L. Smith, N. Dhawan, M. Zhang, P. Abbeel, and S. Levine. Avid: Learning multi-stage tasks via pixel-level translation of human videos, 2020.

[52] P. Sharma, D. Pathak, and A. Gupta. Third-person visual imitation learning via decoupled hierarchical controller, 2019.

[53] H. Xiong, Q. Li, Y.-C. Chen, H. Bharadhwaj, S. Sinha, and A. Garg. Learning by watching: Physical imitation of manipulation skills from human videos. *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 7827–7834, 2021.

[54] K. Schmeckpeper, O. Rybkin, K. Daniilidis, S. Levine, and C. Finn. Reinforcement learning with videos: Combining offline observations with interaction. *CoRR*, abs/2011.06507, 2020. URL https://arxiv.org/abs/2011.06507.

[55] P. Sermanet, K. Xu, and S. Levine. Unsupervised perceptual rewards for imitation learning. *CoRR*, abs/1612.06699, 2016. URL http://arxiv.org/abs/1612.06699.

[56] A. S. Chen, S. Nair, and C. Finn. Learning generalizable robotic reward functions from "in-the-wild" human videos. *ArXiv*, abs/2103.16817, 2021.

[57] M. Sieb, X. Zhou, A. Huang, O. Kroemer, and K. Fragkiadaki. Graph-structured visual imitation. In *Conference on Robot Learning*, 2019.

[58] S. Kumar, J. Zamora, N. Hansen, R. Jangir, and X. Wang. Graph inverse reinforcement learning from diverse videos. In *Conference on Robot Learning*, 2022.

[59] S. Bahl, A. Gupta, and D. Pathak. Human-to-robot imitation in the wild. *ArXiv*, abs/2207.09450, 2022.

[60] T. Yu, P. Abbeel, S. Levine, and C. Finn. One-shot hierarchical imitation learning of compound visuomotor tasks, 2018.

[61] C. Wang, L. J. Fan, J. Sun, R. Zhang, L. Fei-Fei, D. Xu, Y. Zhu, and A. Anandkumar. Mimicplay: Long-horizon imitation learning by watching human play. *ArXiv*, abs/2302.12422, 2023.

[62] M. Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. *Advances in neural information processing systems*, 26, 2013.

[63] 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.

[64] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[65] I. Kostrikov, D. Yarats, and R. Fergus. Image augmentation is all you need: Regularizing deep reinforcement learning from pixels. *arXiv preprint arXiv:2004.13649*, 2020.

[66] P. Sermanet, C. Lynch, Y. Chebotar, J. Hsu, E. Jang, S. Schaal, and S. Levine. Time-contrastive networks: Self-supervised learning from video, 2018.

[67] S. Nair, A. Rajeswaran, V. Kumar, C. Finn, and A. Gupta. R3m: A universal visual representation for robot manipulation, 2022.

[68] A. v. d. Oord, Y. Li, and O. Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

[69] J. Ho, A. Jain, and P. Abbeel. Denoising diffusion probabilistic models. *ArXiv*, abs/2006.11239, 2020.

[70] A. Gupta, V. Kumar, C. Lynch, S. Levine, and K. Hausman. Relay policy learning: Solving long-horizon tasks via imitation and reinforcement learning, 2019.

[71] F. Ebert, Y. Yang, K. Schmeckpeper, B. Bucher, G. Georgakis, K. Daniilidis, C. Finn, and S. Levine. Bridge data: Boosting generalization of robotic skills with cross-domain datasets, 2021.