Appendix

A. Technical Specifications

General Details. All our model training is distributed over NVIDIA 8×A100-80GB (SXM). In all experiments, we employ the ViT-B backbone and to uphold rigorous privacy preservation, we develop conduct training without pre-trained weights from ImageNet. We pre-train VideoMAE for 200 epochs using a tube masking technique that masks 90% of image patches to enhance learned video representations. After pre-training, we remove the VideoMAE decoder, retaining only the encoder. For label alignment, we conduct supervised pre-training for 50 epochs using the same subset of 150 Kinetics classes as SynAPT [36]. For downstream evaluation, we fine-tune (FT) the entire network or train a linear probe (LP) for 30 epochs. Both steps use video inputs as 4D tensors (C, T, H, W), with $C = 3$ (RGB channels), $T = 16$ frames, and spatial dimensions H and W as the video input is resized to 224×224 and normalized.

Table 5. **Summary of Training Details**.

General Specifications	
GPU Configuration	NVIDIA 8×A100-80GB (SXM)
Model Backbone	ViT-B (12 encoder blocks, 768 emb. dim.; 4 decoder blocks, 384 emb. dim.)
Input Tensor Shape	$3 \times 16 \times 224 \times 224$
Normalization	$\mu = [0.485, 0.456, 0.406], \sigma = [0.229, 0.224, 0.225]$
Self-Supervised Pre-Training	
Pre-training Method	VideoMAE
Epochs	200 (10 warm-up)
Masking Strategy	Tube, ratio = 0.9
Batch Size	128
Patch Size	$2 \times 16 \times 16$
Loss Function	MSE
Optimizer	AdamW
Learning Rate (Max)	0.0008
Learning Rate Scheduler	Cosine
Supervised Label Alignment	
Epochs	50 (6 warm-up)
Loss Function	Cross-Entropy
Optimizer	AdamW
Learning Rate (Max)	0.002
Downstream Evaluation	
Adjustment Epochs (FT or LP)	30

Step 1. During self-supervised pre-training, we use mean squared error (MSE) pixel reconstruction loss between the original and reconstructed frames. The batch size is 128, and patch sizes are $2 \times 16 \times 16$. We use the AdamW optimizer [49] with a cosine learning rate scheduler [48], a maximum learning rate of 0.0008, and a 10-epoch warm-up.

Step 2. For supervised label alignment, we add a final linear head to the ViT-B model for supervised training. The model shares an encoder with distinct linear classifiers for each dataset, using cross-entropy loss to measure discrepancies between the predicted and actual action categories. We train the model for 50 epochs with the AdamW optimizer and a cosine rate scheduler with a maximum learning rate of 0.002 and a 6-epoch warm-up.

B. Dataset and Methodology Considerations

B.1. Modified Clustering Operation and Definitions

The K-NEXUS clustering operation, $\kappa(\bar{E}, K)$, aims to minimize the within-cluster bias and pairwise similarity between class embeddings. The clustering operation seeks to minimize the following objective, adapted from [43], which incorporates both bias and pairwise similarity:

$$\arg\min_{\omega} \sum_{k=1}^{K} \left(\sum_{\substack{\bar{\mathbf{e}}_i, \bar{\mathbf{e}}_j \in \omega_k \\ i \neq j}} \mathcal{F}(\bar{\mathbf{e}}_i, \bar{\mathbf{e}}_j) + \sum_{\bar{\mathbf{e}} \in \omega_k} \mathcal{B}(D, \bar{\mathbf{e}}) \right)$$

Here, $\mathcal{B}(D, \bar{\mathbf{e}})$ is the bias measurement for a dataset D using class embedding $\bar{\mathbf{e}}$ and is defined as:

$$\mathcal{B}(D, \bar{\mathbf{e}}) = \ln(\mathcal{M}(D, \bar{\mathbf{e}})) - \ln(\mathcal{M}_{\text{chance}})$$

where $\mathcal{M}(D, \bar{\mathbf{e}})$ represents the performance of the representation $\bar{\mathbf{e}}$ on dataset D, and $\mathcal{M}_{\text{chance}}$ is the performance at the chance level, defined as:

$$\mathcal{M}_{\text{chance}} = \min_{\bar{\mathbf{e}}} \mathcal{M}(D, \bar{\mathbf{e}})$$

Pairwise Similarity Calculation. For each class embedding $\bar{\mathbf{e}}_i$, we calculate the average pairwise similarity with all other class embeddings $\bar{\mathbf{e}}_j$ (where $i \neq j$). Let $\mathcal{F}(\bar{\mathbf{e}}_i, \bar{\mathbf{e}}_j)$ represent the similarity function (e.g., cosine similarity or entropy) between embeddings i and j. The average similarity for class c_i is defined as:

$$M_i = \frac{1}{C-1} \sum_{\substack{j=1 \\ j \neq i}}^{C} \mathcal{F}(\bar{\mathbf{e}}_i, \bar{\mathbf{e}}_j)$$

where C is the total number of classes. This value M_i represents how similar class c_i is to all other classes in the dataset.

Adjusted Bias Calculation. We adjust the bias B_i for class c_i by comparing the pairwise similarity M_i to a baseline chance value M_{chance}. The adjusted bias is given by:

$$B_i = \ln(M_i) - \ln(M_{\text{chance}})$$

This bias B_i accounts for the relationships between class c_i and other classes, and it is used to refine the bias measurements in the clustering process.

Centroid Calculation. The centroid $\bar{\mathbf{e}}_k$ of cluster ω_k is defined as the mean of the bias measurements of all points in ω_k:

$$\bar{\mathbf{e}}_k = \frac{1}{|\omega_k|} \sum_{\bar{\mathbf{e}} \in \omega_k} \mathcal{B}(D, \bar{\mathbf{e}})$$

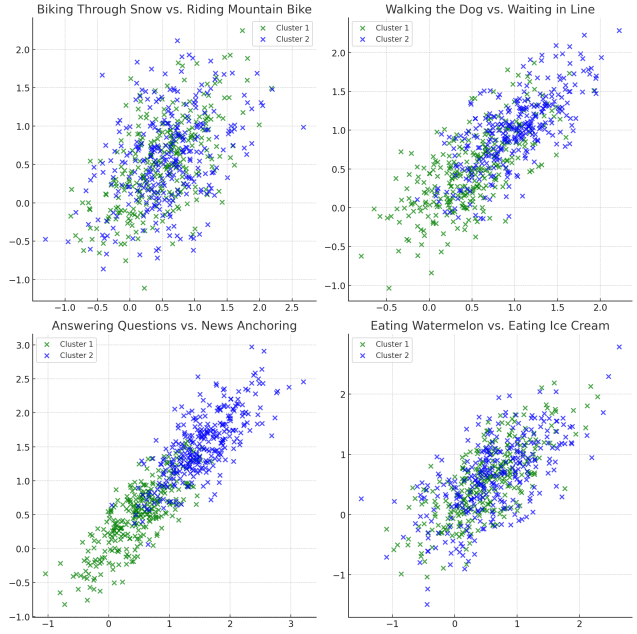
Clustering Steps. This involves the following iterative steps:

1. **Assignment step.** Assign each point $\bar{\mathbf{e}}_i$ to the cluster with the nearest centroid based on both the pairwise similarity and bias measurements:

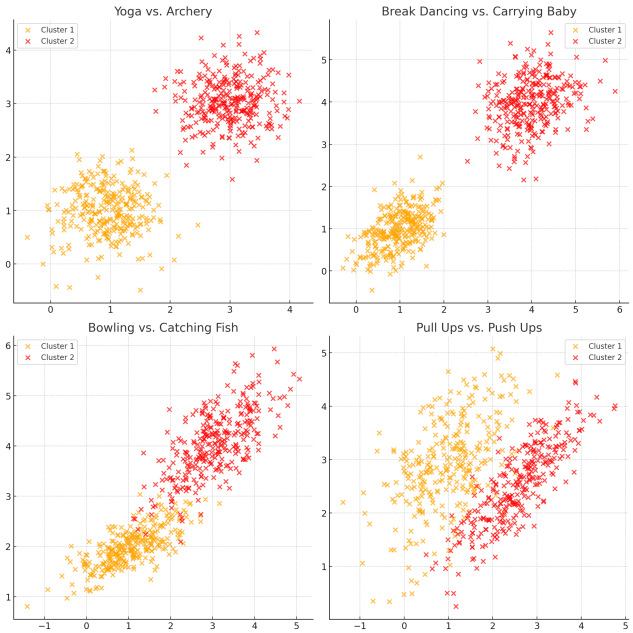
$$\omega_k^{(t+1)} = \left\{ \bar{\mathbf{e}}_i : \mathcal{B}(D, \bar{\mathbf{e}}_k^{(t)}) + \sum_{j \neq i} \mathcal{F}(\bar{\mathbf{e}}_i, \bar{\mathbf{e}}_j) \leq \mathcal{B}(D, \bar{\mathbf{e}}_j^{(t)}) + \sum_{l \neq j} \mathcal{F}(\bar{\mathbf{e}}_j, \bar{\mathbf{e}}_l), \forall j = 1, 2, \ldots, K \right\}$$

2. **Update step.** Calculate the new centroids for each cluster:

$$\bar{\mathbf{e}}_k^{(t+1)} = \frac{1}{|\omega_k^{(t+1)}|} \sum_{\bar{\mathbf{e}}_i \in \omega_k^{(t+1)}} \mathcal{B}(D, \bar{\mathbf{e}}_i)$$



(a) **Example class clusters excluded by K-NEXUS**. The algorithm is designed to identify and exclude categories with overlapping visual cues or semantically broad definitions. It systematically excludes classes prone to ambiguity or high overlap within the embedding space (e.g., "answering questions" vs. "news anchoring" or "biking through snow" vs. "riding a mountain bike"). K-NEXUS considers these as "fine-grained" categories. Otherwise, visually distinct and contextually unique categories (e.g., "archery," "yoga") are retained.



(b) **Example class clusters included by K-NEXUS**. The algorithm is designed to identify and include categories with clearly separable visual cues or semantic definitions. It systematically includes classes that are easy to discretize within the embedding space (e.g., "yoga" vs. "archery" or "bowling" vs. "catching fish"). K-NEXUS considers these as "coarse-grained" categories. Otherwise, visually similar and contextually related categories (e.g., "eating watermelon," "eating ice cream") are discarded.

Figure 3. **Examples of class clusters identified and processed by K-NEXUS.**

Furthermore, the optimization problem to select a subset of classes from the original dataset, as laid out in [43], presents an exponential time complexity of $\mathcal{O}(2^n)$. It is possible to converge to a solution for the selection of a small number of classes. However, it lacks feasibility for our case ($K = 150$ to obtain the Kinetics-150 dataset). Our K-NEXUS approach, converges while having the time complexity of $\mathcal{O}(n \cdot k \cdot t)$, where n is the number of classes, k is the number of clusters, t is the number of update steps. Thus, we are able to perform class sampling for larger values with a linear time complexity.

In this work, we consider the K-NEXUS-selected classes as "coarse-grained" because they represent distinct, well-separated actions that rely less on subtle pose variations or fine contextual cues. These categories are designed to evaluate whether the our proposed framework can effectively learn high-level action semantics without relying on scene or background context. In contrast, "fine-grained" classes involve subtle distinctions, such as variations in hand positioning or object interactions, which present challenges even for fully-supervised models trained on real videos. For this reason, we classify the remaining 250 classes as "fine-grained." Our focus is not on hierarchical annotations or subtle interclass differences across datasets, but rather on the model's ability to handle categories with varying reliance on pose-level distinctions versus scene or temporal context. K-NEXUS classes are intentionally chosen to represent distinct, easily separable actions that primarily depend on human pose, making them coarse-grained for privacy-preserving mesh analysis. See Appendix C for more.

B.2. SMPL Failures

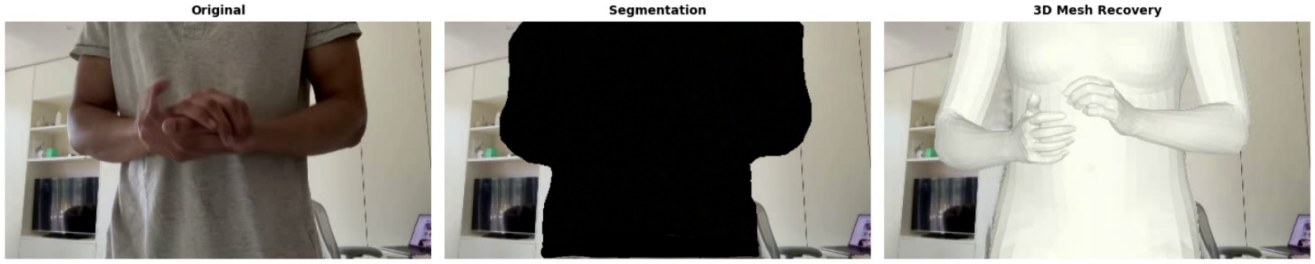


Figure 4. **Only Segmentation Fails.** The action of clapping is visually hidden when using only a segmentation mask but efficiently maintained using 3D mesh recovery post masking and inpainting.

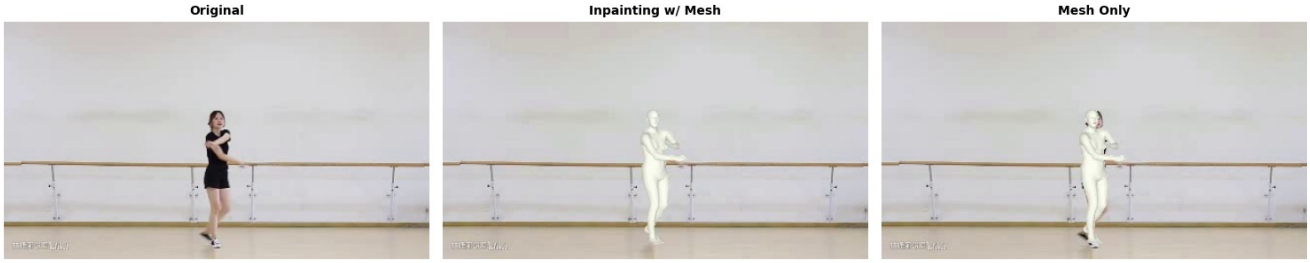


Figure 5. **Only Mesh Recovery Fails.** Employing only meshes exposes a significant chunk of the woman's face, leading to privacy leakage. Private-MO, which incorporates both inpainting and 3D meshes, preserves privacy.

Figure 6. **Failure Cases.** SMPL mesh augmentation without M2M suffers when human joints are occluded. The pottery wheel, music stand, and drums are partially obscured by the superimposed mesh, demonstrating the challenges in handling occlusions within the scene.

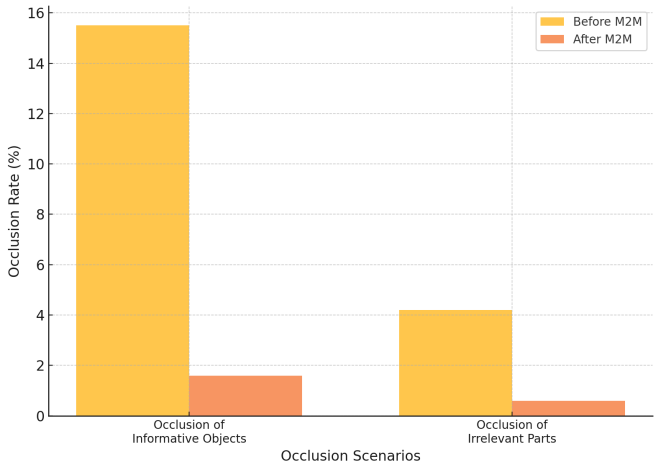


Figure 7. **MORTAL2MESH is occlusion aware.** In our investigation, we had challenges related to occlusions (Figure 6) using just SMPL meshes. To quantify this issue, we manually reviewed 20 randomly selected videos per class from Kinetics-150. Our findings indicated that occlusion-related difficulties were present in 15.5% of the videos. In these cases, the occlusions involved informative objects or backgrounds that contributed to learned features and supervisory signals. An additional 4.2% of cases also had occlusions; however, these did not involve the occlusion of significant objects or scenes essential to the video's labels (i.e., potentially irrelevant parts of the video were occluded, not the major components). Upon using the M2M occlusion-aware augmentation framework (Figure 2), both occlusion rates fell to 1.6% and 0.6% respectively (with 100% interrater reliability).

C. Fine-grained vs. Coarse-grained Action Classification

Table 6. **Comparison of fine-grained and coarse-grained classification performance on Kinetics.** F-scores are reported as the average of mean cluster F-scores for fine-grained classification and as a simple mean F-score for coarse-grained classification. Refer to the end of Appendix B.1 for definitions on what is considered "fine-grained" and "coarse-grained" in this context.

Method	Fine-grained	Coarse-grained
Private-MO w/ K-NEXUS	53.3 ± 0.12	**76.9**
Private-MO	**69.5 ± 0.08**	75.7

We specifically investigated the performance of our approach on both fine-grained and coarse-grained classification tasks. Given that K-NEXUS is designed to curate a set of classes that are coarse-grained, it was crucial to examine how well our model generalizes across fine-grained action classes within these broader categories. To this end, we clustered the Kinetics-400 classes into 150 coarse groups using K-NEXUS and then evaluated the accuracy within these clusters, where the classes are fine-grained. For the fine-tuning phase, we sampled 10% of these clusters (15 in total, filtering for clusters with only one class) and separately fine-tuned our core model on these selected clusters. We report the top-1 mean F-score to account for class imbalance, providing a more nuanced view of the model's performance.

We saw a noticeable performance drop in fine-grained classification when using K-NEXUS splits, with the F-score (53.3). This drop highlights the inherent challenge of fine-grained classification under the K-NEXUS framework. In contrast, when we repeated the experiments using a model pre-trained on random splits [73], the fine-grained classification achieved a significantly higher F-score of 69.5. This suggests that while K-NEXUS is beneficial for coarse-grained classification (F-score > 75), it will introduce limitations for fine-grained tasks. However, it is important to note that for coarse-grained classification, the top-1 F-score on the Kinetics-400 test set for the 150 classes was greater when using K-NEXUS splits compared to random splits. This finding underscores the strength of K-NEXUS in reducing class bias and enhancing generalization across diverse action categories, albeit with some trade-offs in fine-grained classification scenarios.

D. More Evaluations: Genders, Colors, Feature Learning, Run-time, and Framework Ablations

D.1. To Gender or Not To Gender?

Table 7. **Gender-Action Bias Analysis**. We show the performance on gender-biased tasks using real human data vs. 3D meshes. Neutral meshes achieve the highest average accuracy, demonstrating effective mitigation of gender-action bias.

Method	Women in Men-Biased (FT)	Men in Women-Biased (FT)	Mean
VideoMAE w/ real humans	81.4	78.8	80.1
Private-MO w/ male meshes	82.3	**83.3**	82.8
Private-MO w/ female meshes	**83.4**	82.5	82.9
Private-MO w/ neutral meshes	83.2	83.1	**83.1**

In this section, we analyze the impact of using 3D meshes on gender-action bias in action recognition tasks[3]. Specifically, we compare the performance of a model trained on real human data from Kinetics-150. with those trained on our augmented meshed data (M2M Kinetics). We conduct experiments on a specifically curated split of the Kinetics dataset in which women perform male-dominated tasks and men perform female-dominated tasks. The results are summarized in Table 7. Training on real human data revealed significant gender-action bias, with lower performance on "men in women-biased" tasks compared to "women in men-biased" tasks. In contrast, models trained on 3D meshes showed improved performance. Male meshes increased accuracy for "men in women-biased" tasks, while female meshes for "women in men-biased" tasks. Neutral meshes performed consistently well across both subsets. Overall, using 3D meshes outperformed the real human data approach, with higher average scores across all mesh-based methods. This indicates that 3D meshes help mitigate gender-action bias by offering a gender-agnostic representation.

Differences Between Mesh Types. The male, female, and neutral meshes differ primarily in their body shape and proportions, which are modeled to reflect biological and anatomical differences between the genders. The male and female meshes are gender-specific and trained on data tailored to their respective shapes, providing accurate body proportions and capturing gender-specific features like broader shoulders for males or wider hips for females. The neutral mesh is designed to be a compromise between male and female body shapes, enabling its use when the gender is unknown or ambiguous.

How Biased Classes were Selected. We chose women and male-biased classes based the default SMPL-X fitting method as it adapts to the human form within the real data. For instance, if a female human is seen in the video frame, the mesh overlayed will be of the female type. Upon manual inspection, during the process we undertook in Figure 7, we looked at the flipped and neutral cases too just by changing the gender parameter of the mesh for such mesh-fitted videos. Hence, this allowed us to easily categorize classes within the context of "Women in Men-Biased" or "Men in Women-Biased".

D.2. Colored Mesh Ablations: Effects on Accuracy & Privacy

While our primary experiments utilized a default *white* mesh, we also investigated whether changing the color of the SMPL-X meshes could further influence both action-recognition performance and privacy. This inquiry stemmed from two key observations: (i) in privacy benchmarks like VISPR [16], a white-colored mesh might still be misinterpreted as a specific *skin tone*, and (ii) a conspicuously colored mesh could help the model more easily detect human pose or movement in cluttered videos. We retained the same M2M augmentation pipeline (Figure 2) but varied the final mesh color from the default white mesh (baseline) to beige (a hue closer to human skin tone) and rainbow (a vivid, high-contrast gradient pattern). All other training hyperparameters remained constant (Appendix A). We then measured: (1) Top-1 accuracy on downstream benchmarks such as UCF101; (2) Privacy leakage via approximate attribute-inference ability on the VISPR1 subset [16], reported as cMAP (lower cMAP = less privacy leakage).

[3] See Appendix D for more details on this experiment. We used neutral meshes in all our main experiments.

Table 8. **Colored Mesh Experiments.** Action recognition is evaluated on UCF101 (Top-1 FT). Privacy is assessed via cMAP on VISPR1.

Mesh Color	UCF101 (FT)	cMAP (VISPR1)
White (Default)	93.2	33.9
Beige	93.0	34.1
Rainbow	93.5	33.5

Table 8 summarizes the outcomes. Notably, a more conspicuous mesh color can *slightly enhance* action recognition accuracy, while a more skin-like hue (beige) appears to degrade performance and privacy marginally. Switching to a beige tone (closer to real skin) yielded a minor *drop* in accuracy. We also observed a small increase in cMAP on VISPR1 (from 33.9 to 34.1), suggesting that classifiers may still latch onto subtle color cues, undermining strict anonymity. A vibrant, highly contrasted mesh slightly *improved* UCF101 accuracy (from 93.2% to 93.5%). Qualitatively, we noticed that a bright mesh can "stand out" more against the background, possibly aiding the model in focusing on poses and motion. Concurrently, its cMAP dipped to 33.5, implying marginally better privacy-preservation than default white. Our findings underscore an interesting *privacy-versus-utility* trade-off. *Skin-like* hues (beige) may inadvertently restore demographic cues or at least lead a classifier to speculate about such attributes. *Artificially bold* coloring (rainbow) can further anonymize videos and, in some cases, even offer a small performance boost by clearly differentiating the human shape from the scene. Therefore, in privacy-sensitive scenarios (e.g., VISPR-related requirements), using meshes in *unmistakably artificial* colors (rather than human skin approximations) appears advantageous. Beyond reducing any residual attribute leakage, these distinct hues may also help models more rapidly learn action cues, potentially closing the gap to real-human baselines even further.

D.3. Runtime & Computational Analysis

Although Private-MO is primarily designed as an *offline* data-curation pipeline, rather than a real-time edge solution, it is still important to assess its computational overhead. In Table 9, we report the approximate FLOPs and per-video processing times for each major step in our M2M augmentation (masking, inpainting, mesh recovery) as well as the VideoMAE training. All results were measured on an 8×A100-80GB cluster with distributed inference/training. Times can vary depending on implementation details or hardware configurations, but these estimates provide a practical sense of overall cost.

Inference (Video Preprocessing). We use the Segment Anything Model (SAM) [37] to generate dense masks of all objects/people in each frame. Then an optical-flow-based method [44] to fill human-occluded regions after masking. Finally, we employ OSX [45] to estimate pose parameters and generate the 3D mesh. This step is repeated for each video frame to ensure accurate mesh overlay.

Training (VideoMAE). We detail both the FLOPs required to train a VideoMAE model on the augmented (*M2M*) data and a typical per-video time to pass through one epoch. While self-supervised pre-training is more expensive overall, it is a one-time cost. Once the anonymized, mesh-overlaid dataset is prepared, it can be reused for multiple training runs or model architectures.

Table 9. **Approximate FLOPs and per-video runtime.** FLOPs are measured per *single frame*. The video runtime assumes a *distributed* (multi-GPU) setup processing a 10-second clip at 25 fps (250 frames). Inpainting and segmentation remain the most time-consuming parts of our pipeline, but they need only be done once per dataset.

Component	Operation	FLOPs / Frame (G)	Runtime per 10 s Video (s)
SAM	Dense segmentation	792	1.59
E^{2}FGVI	Frame inpainting	293	0.57
OSX (SMPL-X)	Pose parameter fitting	≈ 10	0.02
	Training		
VideoMAE	SSL action-recognition pre-training	≈ 1693	3.42

Implications. Because preprocessing (segmentation → inpainting → mesh fitting) is more computationally expensive than standard data loading, Private-MO is best suited for *offline* anonymization; e.g., preparing a private, mesh-based dataset on sufficiently powerful hardware before any downstream usage. Once anonymized, the newly created M2M-augmented data can be re-used across many training runs, minimizing repetitive cost. If desired, lighter-weight alternatives exist (e.g., Navier-Stokes inpainting [3, 32] instead of E^2FGVI), albeit with minor accuracy drops. Overall, the table shows that while there is some overhead to preserve privacy and reduce demographic bias, this cost can be amortized over multiple uses of the anonymized dataset.

D.4. A "One-Size-Fits-All" Approach for SSL Video Pretraining

To demonstrate that Private-MO can flexibly integrate into existing self-supervised pipelines, we tested *three* MAE-style SSL approaches, Space-time MAE, VideoMAE, and VideoMAE v2, across *two* ViT backbones (ViT-S, ViT-B). Table 10 compares training each method on either *real-human Kinetics* or *M2M Kinetics* (our mesh-based anonymized variant) and then evaluating on five downstream tasks: UCF101, HMDB51, Diving48, IkeaFA, and UAV-Human. Both full finetuning (FT) and linear probing (LP) accuracies are reported, along with a *Realism Gap* column indicating how close M2M Kinetics performance is to the real video data baseline.

Table 10. **Universal Integration of Private-MO with Various MAE Methods.** Each row pairs one SSL method (Space-time MAE, VideoMAE, or VideoMAE v2) and backbone (ViT-S or ViT-B) with either real Kinetics or M2M Kinetics. We list top-1 accuracy (%) for five downstream tasks under FT/LP. The last column measures the "Realism Gap" (difference from real Kinetics). A small negative gap (e.g. -0.2%) means M2M Kinetics is very close to or even exceeding real video data baselines.

SSL Method	Pretraining	Backbone	UCF101 (FT / LP)	HMDB51 (FT / LP)	Diving48 (FT / LP)	IkeaFA (FT / LP)	UAV-Human (FT / LP)	Mean (FT / LP)	Realism Gap (FT / LP)
Space-time MAE	Kinetics	ViT-S	91.0 / 89.2	71.6 / 68.0	64.6 / 19.4	70.3 / 56.9	32.9 / 13.2	66.1 / 49.3	0 / 0
		ViT-B	92.0 / 90.1	72.4 / 68.8	65.3 / 19.6	71.1 / 57.5	34.6 / 13.6	67.1 / 49.9	0 / 0
	M2M Kinetics	ViT-S	91.0 / 88.7	70.9 / 67.6	64.4 / 19.2	69.6 / 56.8	33.8 / 14.0	65.9 / 49.3	−0.2 / −0.0
		ViT-B	91.8 / 89.6	71.5 / 68.2	65.0 / 19.4	70.2 / 57.3	34.1 / 14.1	66.5 / 49.7	−0.6 / −0.2
VideoMAE	Kinetics	ViT-S	92.4 / 90.5	72.7 / 69.0	65.6 / 19.7	71.4 / 57.8	33.4 / 13.4	67.1 / 50.1	0 / 0
		ViT-B (X)	93.4 / 91.5	73.5 / 69.8	66.3 / 19.9	72.2 / 58.4	34.8 / 13.8	68.0 / 50.7	0 / 0
	M2M Kinetics	ViT-S	92.4 / 90.1	71.9 / 68.6	65.4 / 19.5	70.7 / 57.7	34.3 / 14.2	66.9 / 50.0	−0.2 / −0.1
		ViT-B (X)	93.2 / 90.9	72.6 / 69.2	66.0 / 19.7	71.3 / 58.2	34.6 / 14.3	67.5 / 50.5	−0.5 / −0.2
VideoMAE v2	Kinetics	ViT-S	92.6 / 90.8	72.9 / 69.2	65.8 / 19.7	71.6 / 57.9	33.5 / 13.4	67.3 / 50.2	0 / 0
		ViT-B	94.3 / 92.4	74.2 / 70.5	67.0 / 20.1	72.9 / 59.0	35.1 / 13.9	68.7 / 51.2	0 / 0
	M2M Kinetics	ViT-S	93.2 / 90.9	72.6 / 69.2	66.0 / 19.7	71.3 / 58.2	34.6 / 14.3	67.5 / 50.5	0.2 / 0.3
		ViT-B	94.6 / 92.3	73.7 / 70.2	67.0 / 20.0	72.4 / 59.1	35.1 / 14.5	68.6 / 51.2	−0.1 / 0.0

Preserved Performance. For every method/backbone pair, switching from real-human Kinetics to M2M Kinetics *rarely* incurs more than 1% absolute drop. In some instances (*e.g.*, VideoMAE v2 w/ ViT-S), the model actually does *slightly better* on M2M-augmented data (see the $+0.2\%$ / $+0.3\%$ row).

Consistent Gains Across Tasks. The advantage of M2M holds across both "scene-object-biased" tasks (e.g., UCF101) and more challenging, specialized datasets like Diving48 or UAV-Human.

Universal Integration. Because Private-MO anonymizes the *dataset* itself (without modifying the model architecture or training objective), it can serve as a drop-in privacy solution for any modern SSL approach, as evidenced here by Space-time MAE, VideoMAE, and VideoMAE v2 all maintaining strong performance. From these experiments, we see that *mesh-based anonymization* can function as a "one-size-fits-all" enhancement to existing video-pretraining pipelines. Despite removing human identities, M2M Kinetics nearly matches or occasionally *surpasses* the accuracy of real-human pretraining, greatly expanding the applicability of SSL in privacy-sensitive contexts.

D.5. Data Splits

K-NEXUS

- *Included*: abseiling, air drumming, applauding, archery, arm wrestling, arranging flowers, baby waking up, baking cookies, barbequing, bee keeping, belly dancing, bending back, blasting sand, bobsledding, bouncing on trampoline, bowling, breakdancing, brush painting, brushing teeth, bungee jumping, canoeing or kayaking, capoeira, carrying baby, catching fish, celebrating, cheerleading, clapping, clay pottery making, climbing a rope, climbing tree, cooking chicken, crawling baby, crying, dancing ballet, dancing gangnam style, decorating the christmas tree, dodgeball, doing aerobics, doing nails, drawing, drinking, driving car, drop kicking, eating burger, eating ice cream, exercising arm, filling eyebrows, finger snapping, flying kite, folding clothes, garbage collecting, grooming dog, gymnastics tumbling, high jump, hopscotch, hugging, hula hooping,

javelin throw, jogging, juggling balls, jumping into pool, kicking soccer ball, kissing, kitesurfing, laughing, making a cake, making snowman, massaging legs, milking cow, moving furniture, mowing lawn, paragliding, parkour, passing American football (in game), petting animal (not cat), playing accordion, playing badminton, playing basketball, playing cello, playing controller, playing didgeridoo, playing drums, playing flute, playing guitar, playing ice hockey, playing monopoly, playing paintball, playing piano, playing squash or racquetball, playing tennis, playing trumpet, playing violin, playing volleyball, pole vault, presenting weather forecast, pull ups, pumping fist, push up, pushing cart, reading book, riding a bike, riding elephant, riding or walking with horse, riding unicycle, rock climbing, roller skating, running on treadmill, scuba diving, shaking hands, shaking head, shaving head, shooting basketball, shooting goal (soccer), side kick, situp, skateboarding, skiing (not slalom or crosscountry), skiing slalom, snorkeling, snowboarding, spraying, stretching arm, surfing water, sweeping floor, swimming butterfly stroke, sword fighting, taking a shower, tap dancing, testifying, texting, training dog, trimming or shaving beard, tying bow tie, unboxing, using computer, using remote controller (not gaming), walking the dog, washing dishes, water skiing, water sliding, waxing back, waxing eyebrows, welding, wrapping present, wrestling, writing, yawning, yoga, zumba.

- *Excluded*: answering questions, applying cream, assembling computer, auctioning, balloon blowing, bandaging, bartending, beatboxing, bench pressing, bending metal, biking through snow, blowing glass, blowing leaves, blowing nose, blowing out candles, bookbinding, braiding hair, breading or breadcrumbing, brushing hair, building cabinet, building shed, busking, cartwheeling, carving pumpkin, catching or throwing baseball, catching or throwing frisbee, catching or throwing softball, changing oil, changing wheel, checking tires, chopping wood, clean and jerk, cleaning floor, cleaning gutters, cleaning pool, cleaning shoes, cleaning toilet, cleaning windows, climbing ladder, contact juggling, cooking egg, cooking on campfire, cooking sausages, counting money, country line dancing, cracking neck, crossing river, curling hair, cutting nails, cutting pineapple, cutting watermelon, dancing charleston, dancing macarena, deadlifting, digging, dining, disc golfing, diving cliff, doing laundry, dribbling basketball, drinking beer, drinking shots, driving tractor, drumming fingers, dunking basketball, dying hair, eating cake, eating carrots, eating chips, eating doughnuts, eating hotdog, eating spaghetti, eating watermelon, egg hunting, exercising with an exercise ball, extinguishing fire, faceplanting, feeding birds, feeding fish, feeding goats, fixing hair, flipping pancake, folding napkins, folding paper, front raises, frying vegetables, gargling, getting a haircut, getting a tattoo, giving or receiving award, golf chipping, golf driving, golf putting, grinding meat, grooming horse, hammer throw, headbanging, headbutting, high kick, hitting baseball, hockey stop, holding snake, hoverboarding, hurdling, hurling (sport), ice climbing, ice fishing, ice skating, ironing, jetskiing, juggling fire, juggling soccer ball, jumpstyle dancing, kicking field goal, knitting, krumping, laying bricks, long jump, lunge, making a sandwich, making bed, making jewelry, making pizza, making sushi, making tea, marching, massaging back, massaging feet, massaging person's head, mopping floor, motorcycling, news anchoring, opening bottle, opening present, parasailing, passing American football (not in game), peeling apples, peeling potatoes, petting cat, picking fruit, planting trees, plastering, playing bagpipes, playing bass guitar, playing cards, playing chess, playing clarinet, playing cricket, playing cymbals, playing harmonica, playing harp, playing keyboard, playing kickball, playing organ, playing poker, playing recorder, playing saxophone, playing trombone, playing ukulele, playing xylophone, pumping gas, punching bag, punching person (boxing), pushing car, pushing wheelchair, reading newspaper, recording music, riding camel, riding mechanical bull, riding mountain bike, riding mule, riding scooter, ripping paper, robot dancing, rock scissors paper, sailing, salsa dancing, sanding floor, scrambling eggs, setting table, sharpening knives, sharpening pencil, shaving legs, shearing sheep, shining shoes, shot put, shoveling snow, shredding paper, shuffling cards, sign language interpreting, singing, ski jumping, skiing crosscountry, skipping rope, skydiving, slacklining, slapping, sled dog racing, smoking, smoking hookah, snatch weight lifting, sneezing, sniffing, snowkiting, snowmobiling, somersaulting, spinning poi, spray painting, springboard diving, squat, sticking tongue out, stomping grapes, stretching leg, strumming guitar, surfing crowd, swimming backstroke, swimming breast stroke, swing dancing, swinging legs, swinging on something, tai chi, tango dancing, tapping guitar, tapping pen, tasting beer, tasting food, throwing ball, throwing discus, tickling, tobogganing, tossing coin, tossing salad, trapezing, trimming trees, triple jump, tying knot (not on a tie), tying tie, unloading truck, using segway, vault, waiting in line, washing feet, washing hair, washing hands, watering plants, waxing chest, waxing legs, weaving basket, whistling, windsurfing.

Gender Splits (Kinetics-150)

- *Female*: arranging flowers, baby waking up, baking cookies, belly dancing, brushing teeth, cheerleading, cooking chicken, crying, dancing ballet, decorating the christmas tree, doing aerobics, doing nails, drawing, exercising arm, filling eyebrows, folding clothes, hugging, kissing, laughing, making a cake, massaging legs, petting animal (not cat), playing piano, shaking head, shaving head, stretching arm, taking a shower, texting, trimming or shaving beard, waxing back, waxing eyebrows, wrapping present, writing, yawning, yoga, zumba.

- *Male*: abseiling, archery, arm wrestling, barbequing, bobsledding, catching fish, climbing a rope, climbing tree, driving car, drop kicking, flying kite, garbage collecting, javelin throw, kicking soccer ball, moving furniture, mowing lawn, paragliding, parkour, passing american football (in game), playing ice hockey, pole vault, pull ups, push up, riding a bike, rock climbing, shooting basketball, shooting goal (soccer), side kick, skateboarding, sword fighting, welding, wrestling.
- *Neutral*: air drumming, applauding, bee keeping, bending back, blasting sand, bouncing on trampoline, bowling, breakdancing, brush painting, bungee jumping, canoeing or kayaking, capoeira, carrying baby, celebrating, clapping, clay pottery making, crawling baby, dancing gangnam style, dodgeball, drinking, eating burger, eating ice cream, finger snapping, grooming dog, gymnastics tumbling, high jump, hopscotch, hula hooping, jogging, juggling balls, jumping into pool, kitesurfing, making snowman, milking cow, playing accordion, playing badminton, playing basketball, playing cello, playing controller, playing didgeridoo, playing drums, playing flute, playing guitar, playing monopoly, playing paintball, playing squash or racquetball, playing tennis, playing trumpet, playing violin, playing volleyball, presenting weather forecast, pumping fist, pushing cart, reading book, riding elephant, riding or walking with horse, riding unicycle, roller skating, running on treadmill, scuba diving, shaking hands, situp, skiing (not slalom or crosscountry), skiing slalom, snorkeling, snowboarding, spraying, surfing water, sweeping floor, swimming butterfly stroke, tap dancing, testifying, training dog, tying bow tie, unboxing, using computer, using remote controller (not gaming), walking the dog, washing dishes, water skiing, water sliding.