Unpaired Game-To-Movie Translation using One-Step Diffusion Models

Mattia Salsi

Abstract

This paper explores the use of one-step diffusion models to perform unpaired image-to-image translation. We utilise LoRA adapters to fine-tune a pre-trained diffusion model using an adversarial approach, and integrate additional conditioning in order to allow the model to localise the most important features to translate.

1 Human Feature Analysis

1.1 Human Patch Extraction

For our first pass over the videos, we utilise YOLO [3] to detect human patches, due to its fast inference performance with minimal accuracy trade-off in comparison to competitors. In order to avoid storing similar frames, we use OpenCV [1] to extract SIFT features [4] and match them to recently stored frames in a lookup-buffer using FLANN matching - a nearest neighbour algorithm. We skip any frames that exceed a given similarity threshold. Collected frames are shown in Figure 1.

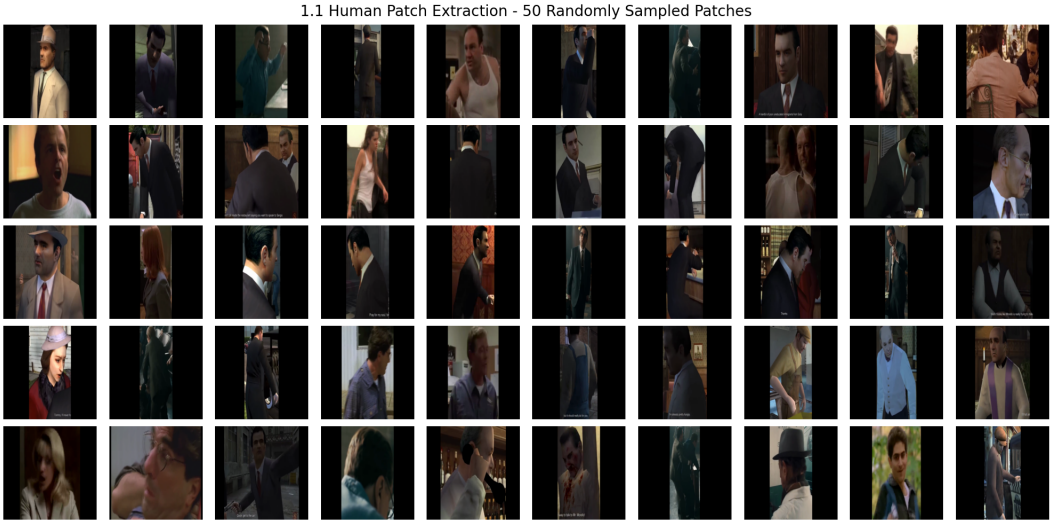

Figure 1: 50 randomly sampled human patches.

1.2 Classification

To classify human patches, we leverage local features extracted using two methods: (i) MTCNN face-matching [8], and (ii) YOLO pose estimation. We find that using both MTCNN and YOLO poses to detect faces helps to mitigate false positives.

YOLO returns 17 pose points, which we use to divide the body into four categories: face, side-profile, torso, and legs. For each category to be detected, we require that some percentage of features are present [Table 1]. Finally, we classify the pose given which body categories are active [Table 2].

YOLO Pose Category	Features Required	Percent Required
Face	Nose, L/R eye, L/R ear	80%
Side-Profile	Nose, L/R eye, L/R ear	60%
Torso	L/R shoulder, L/R ear	50%
Legs	L/R hip, L/R leg, L/R foot	33%

Table 1: A break-down of which pose features are required for a given body category to be detected.

Pose	MTCNN Face	Face	Torso	Legs	Side-Profile
Front H&S	✓	✓	✓		
Back H&S			✓		
Front Full Body	✓	✓	✓	✓	
Back Full Body			✓	✓	
Other					✓

Table 2: A break-down of which body categories are required for a given pose to be detected.

Most patches fall into one of four poses, with the majority of those in 'other' being side-views. Visualisations are shown in Appendix figures [8], [9], [10], [11], and [12].

1.3 Training Data Selection

We perform another pass through our human patches using a YOLO segmentation model in order to extract tighter, pixel-wise bounds on humans. A patch may have multiple humans, in which case we save each as a separate training example.

We then filter each human segmentation by defining three conditions: (i) a minimum illumination, (ii) a maximum blur detected using FFT, and (iii) a minimum patch size.

We apply face detection and pose estimation on each training example, and remove any which return multiple detections. We want each example to strictly contain only one human, to minimise variance and improve the down-stream game-to-movie translation by minimally affecting background pixels.

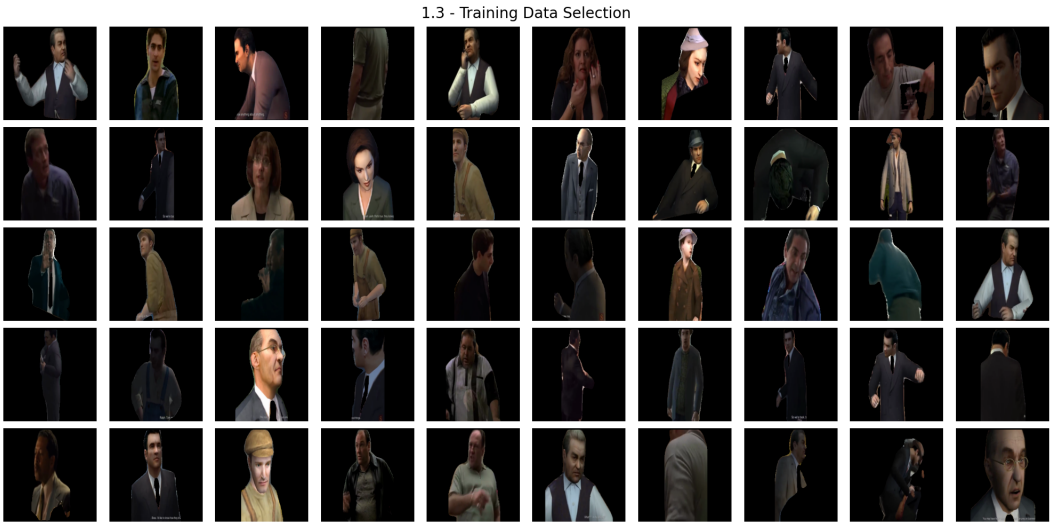

Figure 2: 50 randomly sampled examples from our final training data.

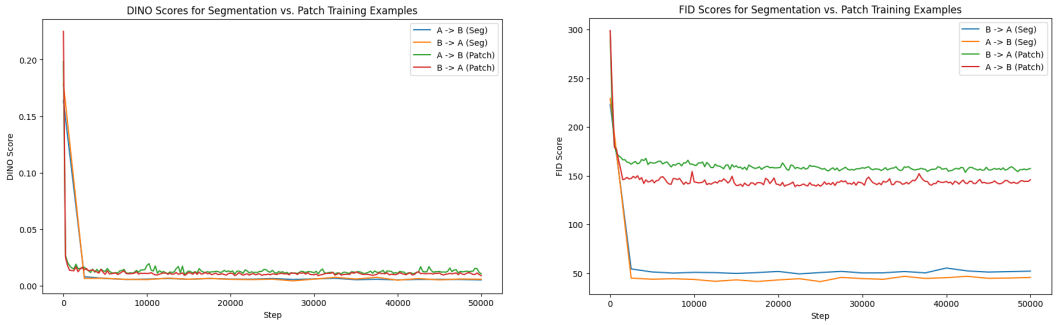

Figure 3: DINO scores (left) and FID scores (right) for two model configurations - one trained on segmentation masks, and the other on bounding boxes. Training with segmentation masks removes background information and therefore yields increased performance for both metrics (lower is better).

2 Real-world Application

2.1 Image Model Deployment

We follow the approach proposed by Parmar *et al.* [5], allowing us to fine-tune SD-Turbo by StabilityAI [6] with significantly less memory requirements.

The technique uses LoRA adapters [2], allowing us to freeze the original model weights and instead train two lower-rank weight matrices, A and B, to modify the output of original weights W as:

$$y = (W + AB)x \tag{1}$$

where x is the input, y is the output, and $A \in \mathbb{R}^{d\times r}$, $B \in \mathbb{R}^{r\times d}$ are the learnable low-rank LoRA matrices with $r \ll d$.

Despite their traditional Markovian sampling procedure, distilled single-step diffusion models p_θ allow us to always predict a clean sample $\hat{\mathbf{x}}_0 = p_\theta(\mathbf{x}_0|\mathbf{x}_T)$, $\mathbf{x}_T \sim \mathcal{N}(0, \mathbf{I})$. It is then possible to fine-tune the model towards image translation using a typical CycleGAN [9] adversarial objective. Though, unlike CycleGAN, we utilise only one generator G, which is conditioned using a prompt c_X or c_Y depending on the translation direction $Y \to X$ m or $X \to Y$.

We find that the Game-to-Movie translation is very effective, with the model adding convincing detail especially to character faces [Figure 4]. The most common failure for Game-to-Movie translation is an incorrect lighting adjustment, which may be caused by our segmentation training examples limiting global information. However, training with segmentation masks leads to better evaluation metrics [Figure 3]. We observe a higher frequency of failure for Movie-to-Game translation due to the low variation in the game patches, leading to the model trying to add artefacts such as shirts and blood where it should not [Figure 5].

2.2 Local Enhancement

We observe that our original model does not suffer from excessive flickering, and therefore choose not to focus on temporal enhancements. Instead, we adapt our model to incorporate the local facial and pose features extracted by MTCNN and YOLO, in order to better localise important structures to augment.

Specifically, we convert MTCNN facial coordinates (x_1, y_1, x_2, y_2) into a map $\mathcal{F}_{\text{face}} \in \mathbb{R}^{256\times256}$ where $\mathcal{F}^{ij}_{\text{face}} = 1$ if $x_1 < i < x_2$ and $y_1 < j < y_2$, and 0 otherwise.

Figure 4: 10 successful translations - 5 from Movie2Game, 5 from Game2Movie.

Figure 5: 10 failed translations - 5 from Movie2Game, 5 from Game2Movie.

Additionally, we create a pose map $\mathcal{F}_{\text{pose}} \in \mathbb{R}^{256\times256}$, where $\mathcal{F}^{ij}_{\text{pose}} = p$, if pixel (i, j) belongs to feature p, for $p \in [1, P]$ of total pose features P.

We concatenate $\mathcal{F}_{\text{face}}$ and $\mathcal{F}_{\text{pose}}$ channel-wise with an image $\mathbf{x} \in \mathbb{R}^{3\times256\times256}$ to get $\tilde{\mathbf{x}} = \mathbf{x} \oplus \mathcal{F}_{\text{face}} \oplus \mathcal{F}_{\text{pose}} \in \mathbb{R}^{5\times256\times256}$.

We create a fusion layer $F(\mathbf{x}) : \mathbb{R}^{5\times256\times256} \to \mathbb{R}^{3\times256\times256}$, made up of two residual blocks, to combine the features back to the original dimensionality, which is then provided to the generator as before.

We compare the performance of the two models using DINO and FID metrics, which measure perceptual similarity between our generated samples and those in the test set. We find that the introduction of extra features improves DINO performance but decreases FID [Figure 7], though we note that DINO provides richer representations by comparing distance using feature maps extracted using Transformers [7].

Perceptually, we find the images generated by the conditional model to be more appealing - mostly due to an increase in colour saturation, especially noticeable on faces which appear more blushed and natural [Figure 6].

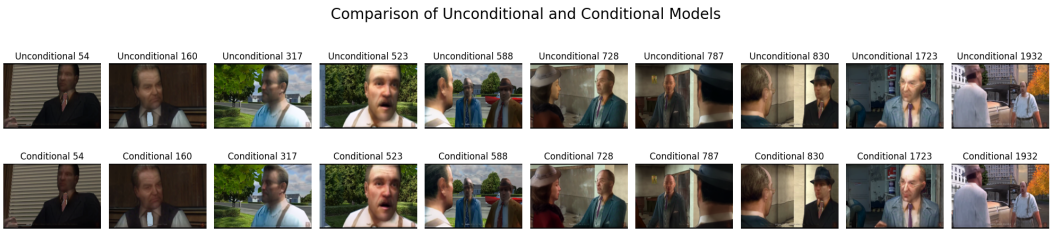

Figure 6: Comparison of 10 translated frames from the unconditional model (top) and conditional model (bottom). The conditional model produces facial features with more vibrant shades.

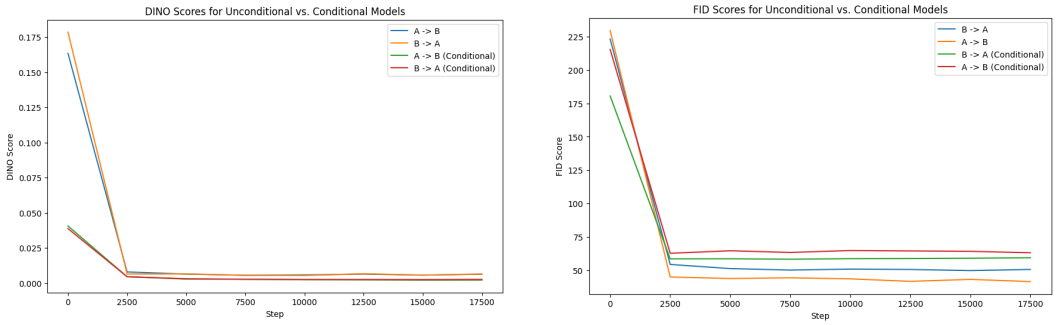

Figure 7: DINO scores (left) and FID scores (right) for two model configurations - one unconditional, and one provided with extra features covering the face and pose of human detections.

References

[1] G. Bradski. "The OpenCV Library". In: *Dr. Dobb's Journal of Software Tools* (2000).

[2] Edward J. Hu et al. *LoRA: Low-Rank Adaptation of Large Language Models*. 2021. arXiv: 2106.09685 [cs.CL].

[3] Glenn Jocher, Ayush Chaurasia, and Jing Qiu. *Ultralytics YOLOv8*. Version 8.0.0. 2023. URL: https://github.com/ultralytics/ultralytics.

[4] David G Lowe. "Object recognition from local scale-invariant features". In: *Proceedings of the seventh IEEE international conference on computer vision*. Vol. 2. Ieee. 1999, pp. 1150–1157.

[5] Gaurav Parmar et al. *One-Step Image Translation with Text-to-Image Models*. 2024. arXiv: 2403.12036 [cs.CV].

[6] Axel Sauer et al. *Adversarial Diffusion Distillation*. 2023. arXiv: 2311.17042 [cs.CV].

[7] Ashish Vaswani et al. *Attention Is All You Need*. 2023. arXiv: 1706.03762 [cs.CL].

[8] Kaipeng Zhang et al. "Joint Face Detection and Alignment Using Multitask Cascaded Convolutional Networks". In: *IEEE Signal Processing Letters* 23.10 (2016), pp. 1499–1503. DOI: 10.1109/LSP.2016.2603342.

[9] Jun-Yan Zhu et al. *Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks*. 2020. arXiv: 1703.10593 [cs.CV].

2.3 Appendix

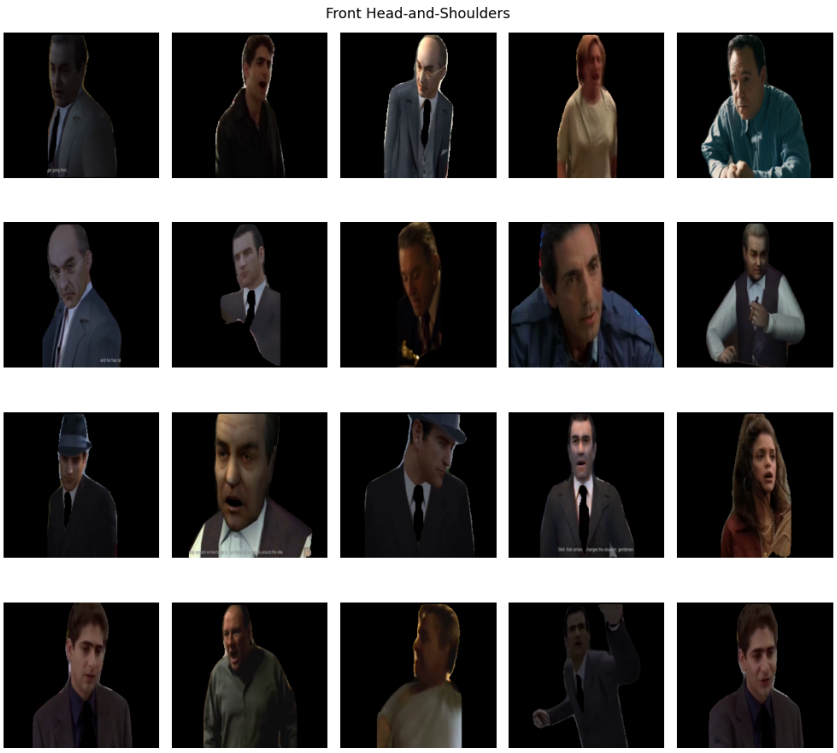

Figure 8: 20 randomly sampled front head-and-shoulders patches.

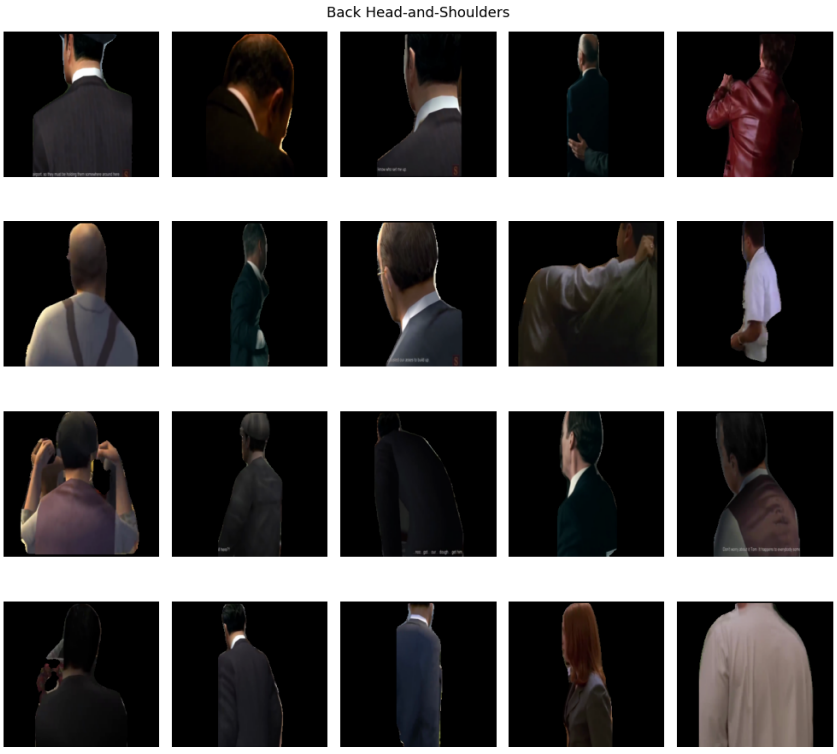

Figure 9: 20 randomly sampled back head-and-shoulders patches.

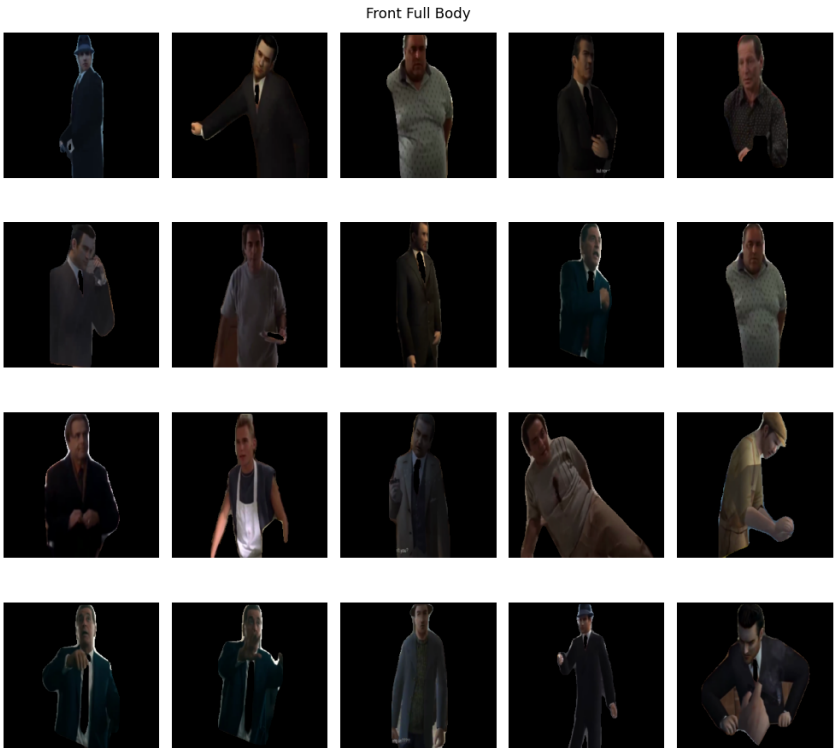

Figure 10: 20 randomly sampled front full-body patches.

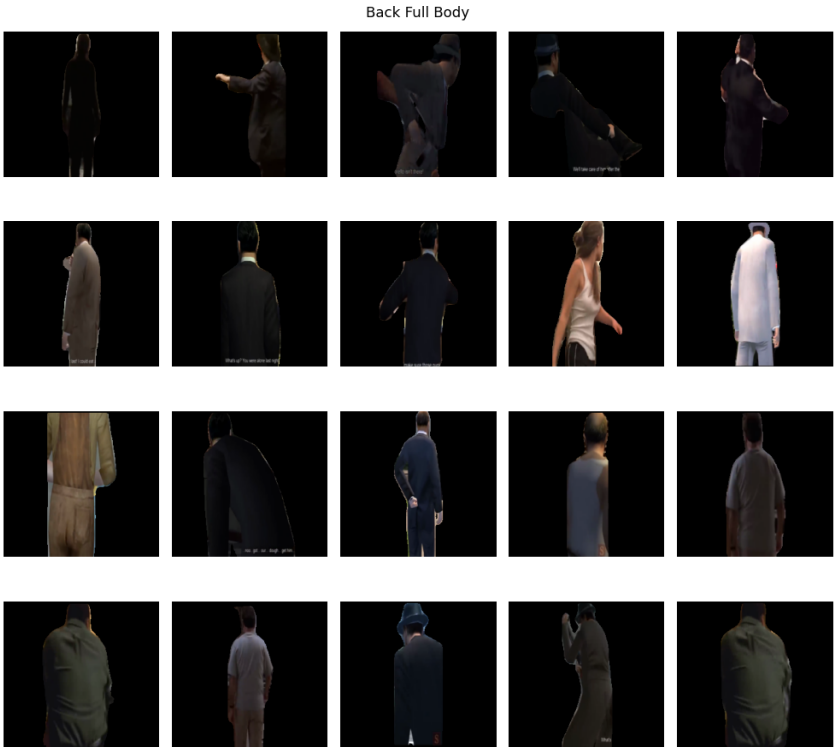

Figure 11: 20 randomly sampled back full-body patches.

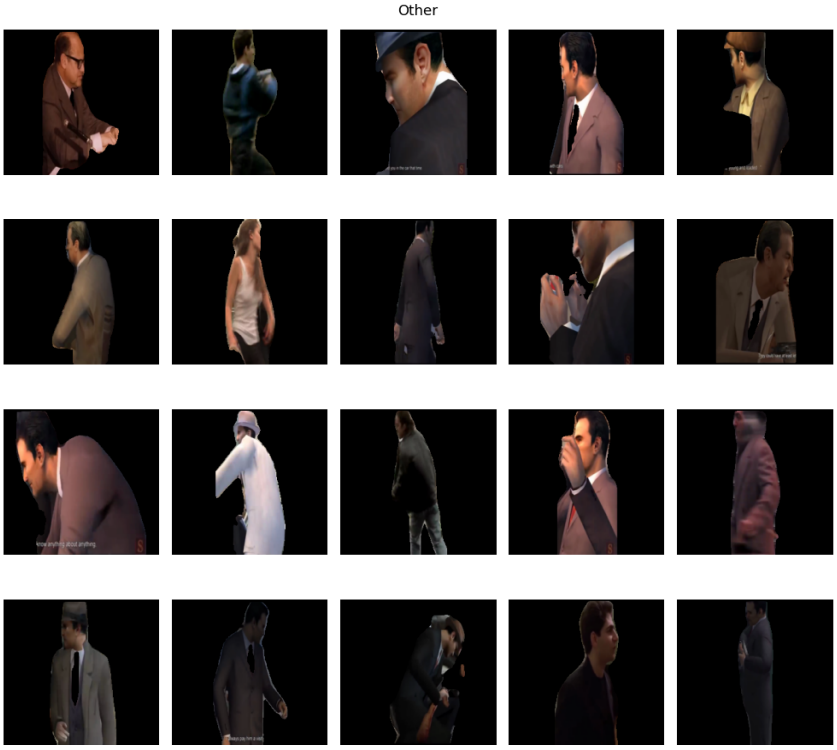

Figure 12: 20 randomly sampled patches with 'other' pose detection.