Are we SMPLy biased?

Identifying ethical biases in Action Recognition

Ana Bălțărețu

Are we SMPLy biased?

Identifying ethical biases in Action Recognition

by

Ana Bălțărețu

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on
2nd of July 2025

Student number: 5008395
Project Duration: September, 2024 - June, 2025
Thesis committee: Dr. J. van Gemert, TU Delft, supervisor
P. Benschop, TU Delft, daily supervisor
Dr. M Skrodzki, TU Delft

TU Delft

Preface

In this project my biggest challenge was on the mental side. I realized that consistent motivation is hard to maintain when working alone on a project for an entire year. Luckily for me, I had a strong network that supported me through this journey.

I would like to thank my parents, Simona and Viorel, for their continuous support during both my Bachelor's and Master's degrees in the Netherlands. Thank you for making it possible for me to be here in the first place, and for allowing me to rest, travel, and to take my time with my studies.

I would like thank all of my friends for the numerous hours that they spent with me either studying on campus, participating in events that we organize, cooking, playing board games, or just for simply hanging out together. In particular, I want to thank my friend Marina for being there for me almost every day in the past year, for keeping both my mental and physical health strong (slay bouldering bro), and for simply checking in on me.

I would also like to thank my mentors, Jan and Pascal, for their consistent and helpful feedback, and for their visible passion for the topic I was working on. Pascal thank you for keeping me hyped up about the project, even when I was critical of the developments. Jan, thank you for sharpening my critical thinking skills during the project, your questions pushed me to think more deeply and articulate my ideas more clearly.

Ana
Delft, June 2025

Contents

1

Introduction

As Action Recognition models are released into real-world applications, they risk perpetuating the hidden biases of their designers, as well as those embedded in the datasets and modeling choices that shape their development. This thesis investigates such biases by looking beyond model accuracy, focusing instead on specific visual attributes that could trigger different treatment for some misrepresented groups, and seeing under what conditions these effects arise.

At the core of this project is Computer Vision, a field focused on enabling machines to extract meaningful information from visual data such as images and videos. Grounded in mathematics, engineering, and artificial intelligence, it aims to replicate aspects of human perception through computational models. In this project, Computer Vision provides both the foundation and the critical lens: we study Action Recognition models not only in terms of performance, but in relation to their robustness, the groups they misrepresent or overlook, and the conditions under which biases emerge. Computer Vision is an inherently interdisciplinary field, bridging perception and computation to explore how machines can replicate or augment human visual understanding. In this thesis, it intersects with Computer Graphics, used to generate synthetic data, and with regulatory concerns, such as those raised by the EU AI Act [1, 2], which emphasizes the need for transparency and fairness, in particular, during the placement on the market of the AI system in the European Union.

This chapter introduces the core concepts that form the foundation of the thesis, each contributing to a deeper understanding of bias in Action Recognition. We begin by exploring concepts of Deep Learning, which forms the backbone of modern Action Recognition models and is central to discussions on model behavior and bias. Computer Graphics is introduced as a key tool for generating synthetic data, enabling controlled experimentation with visual variations. Data Visualization techniques help interpret model performance and reveal hidden patterns of bias. Finally, we reflect on the EU AI Act, framing the societal and ethical implications of this work within emerging regulatory efforts.

After this introductory chapter, we present the scientific paper in which we further explain the problem, describe our approach for answering the research question and we analyze the results of our experiments. The thesis concludes by reflecting on the ethical implications of this project and outlining directions for future research, including insights from other experiments discussed in the additional information section.

1.1. Intro to Deep learning & Neural networks

In this section we go over the fundamental concepts related to training deep neural networks, with the simplest trainable model being the perceptron. We also describe some core network architectures used by the evaluated models, such as convolutional neural networks and transformers.

1.1.1. The perceptron

A perceptron [3] is a fundamental computational unit used in machine learning for classification tasks. It represents a simplified model of a biological neuron by processing inputs, combining them with learned weights, activating or producing an output based on whether a threshold is exceeded. This model separates data into distinct categories by learning from labeled examples and adjusting its parameters (weights and biases) to improve prediction accuracy over time (i.e. train). Perceptrons are the building blocks of neural networks, but when we say "perceptron", we're usually referring to the original single-layer model, composed of the following components:

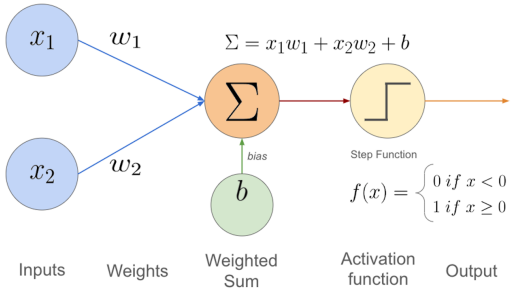


Figure 1.1: A single layer perceptron, inputs (x), parameters (weights w and biases b), summation and activation functions.[1]

- **Input Features (x_i)**: represents a numerical attribute of the input data.
- **Model parameters** are its weights and biases. These parameters are updated during training to find an optimal value.
  - **Weights (w_i)**: Each input feature is assigned a weight indicating its importance in the decision-making process.
  - **Bias (b)**: The bias term shifts the decision boundary, allowing the perceptron to model data not centered around the origin.
- **Summation Function ($\sum$)**: The perceptron computes a weighted sum of the inputs, giving each input different importance: $z = \sum w_i x_i + b$.
- **Activation Function ($f(x)$ or $\phi(x)$)**: This function determines the output based on whether the weighted sum passes a threshold. In the original perceptron, the step function was used to produce binary output. In modern neural networks, activation functions such as ReLU or Sigmoid are used to introduce non-linearity, enabling the network to model more complex patterns.
- **Output (y)**: The output is the result of the activation function and represents the predicted class.

When we refer to a "neural network", we typically mean a series of perceptrons stacked in layers. This layered structure, combined with non-linear activation functions, enables the network to solve complex tasks such as non-linearly separable problems (XOR) and multi-class classification (digit recognition). A typical neural network consists of an input layer, one or more hidden layers, and an output layer. These interconnected neurons form the foundation of modern deep learning, allowing models to learn representations from data, by making use of:

- **Learning Algorithm**: The perceptron adjusts its weights and bias using a learning algorithm, through backpropagation, to minimize prediction errors.
- **Training Data**: varies depending on the task and the domain. In our project we use RGB monocular videos, but the training data can be essentially anything with a numerical representation.
- **Labels**: are generally used to determine whether the output of the model matches with the ground truth, though there are also unsupervised networks which can be trained without labels.

[1] https://muneebsa.medium.com/deep-learning-101-lesson-7-perceptron-f6a698d81be8

1.1.2. Convolutional Neural Networks (CNNs)

Perceptrons treat each pixel as an independent input to the network, while convolutional neural networks evaluate image patches, essentially taking a decision based on a pixel and their neighbors. Instead of learning one weight per edge in the network, CNNs learn filters (or kernels) for each layer in the network, which are applied to the images from previous layers to automatically extract features. By applying multiple such kernels, CNNs transform the image into a lower dimensional representation which is an attempt to retain the relevant information. Convolutions form the basics of modern image neural networks and are used in some of the models we evaluated in the paper, such as SlowFast [4] and X3D [5].

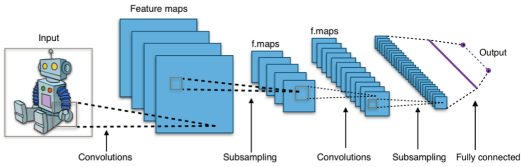


Figure 1.2: Typical CNN architecture[2], showing kernels being applied on an input image, leading to feature maps being created.

1.1.3. Latent space

Similarly to CNNs, other deep learning architectures map inputs to a low-dimensional representation known as the latent space. Ideally, once trained, the network should place similar inputs close together in this space, and in our context, we might expect videos of people performing the same action to cluster near one another. However, this behavior isn't guaranteed, since we don't have control over which features the model uses to organize the latent space. Instead of action similarity, the model might group videos based on camera viewpoint, background, or other visual cues. This uncertainty around what features guide the model's internal organization contributes to the challenge of explainability.

1.1.4. Attention & Transformers

CNNs have been successful in visual tasks due to their ability to extract local features, however, they treat all parts of an image equally and rely heavily on biases such as the influence of neighboring pixels. This makes it difficult for CNNs to capture long-range dependencies between pixels further apart. Attention mechanisms address this limitation by allowing models to prioritize the most relevant parts of the input, mimicking how humans focus on specific areas in a scene. Self-attention relates different positions within a single input sequence to each other. In computer vision, self-attention allows models to capture global context, which is crucial for understanding spatial relationships within an image or temporal dynamics in a video.

Transformers [6] are a network architecture that essentially combines multiple attention layers, initially used in natural language processing. Vision Transformers (ViTs) [7] extend this approach to computer vision by representing images as sequences of patches, demonstrating that CNNs are not a requirement and that transformers alone can perform effectively on image classification tasks. Out of the evaluated models for this project, TC-Clip [8] makes use of attention and MViT [9] relies on vision transformers, further explained in subsection 1.3.2.

[2] https://upload.wikimedia.org/wikipedia/commons/6/63/Typical_cnn.png

1.2. Intro to Computer Graphics

Computer graphics is a field of computer science focused on generating, manipulating, and rendering visual content. It spans from the mathematical principles behind image creation to practical methods that optimize visual presentation.

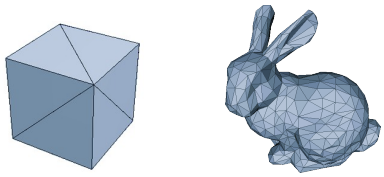


Figure 1.3: Simple cube model and more complex bunny made of triangles.

3D modelling body templates. Every 3D object rendered on screen, from simple cubes to detailed characters, is composed of triangles, Figure 1.3. When rendering, these objects pass through a graphics pipeline that transforms complex geometry into pixels. In our scenes, actors are modeled using the SMPL body model [10] (subsection 1.2.3), which features complex geometry. To render thousands of short videos efficiently, we use Unreal Engine's Movie Render Queue, a high-quality rendering tool that supports batch processing and realistic lighting. Because Unreal handles the low-level rendering internally, we can focus on content creation rather than implementation details of the graphics pipeline.

1.2.1. Textures

In computer graphics, a texture is a two-dimensional image or pattern applied to the surface of a 3D model to provide its visual appearance. While the 3D model defines an object's geometry, texture maps add surface-level detail like color, reflectivity, and roughness in a computationally efficient way. They are fundamental in real-time applications such as video games and interactive simulations, where performance and visual fidelity must be carefully balanced, and they also reduce rendering time when creating complex scenes. A texture is an unwrapped representation of the appearance of a 3D model. Instead of modeling fine surface details (like wrinkles, scratches, or fabric patterns) directly in the mesh, which would require a large number of polygons and increase rendering time, texture maps store this information in lightweight 2D images. For instance, Figure 1.4 shows a flattened 2D texture of a Rubik's cube. Through a process called UV mapping, these 2D textures are projected onto 3D surfaces, by aligning each point on the texture with corresponding coordinates on the model. There are several types of textures used in 3D graphics, each encoding different visual properties of an object, including:

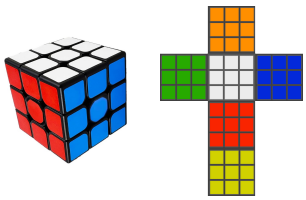


Figure 1.4: Model of a cube vs its un-wrapped texture map.

- *Diffuse (or Albedo) textures* define the base color of a surface, without lighting or shading information. This is the primary type of texture used in our project, where we focus on modifying the skin tone of characters.
- *Normal maps* simulate small surface details like by altering how light interacts with a surface, without increasing geometric complexity.
- *Specularity maps* control how shiny or reflective different parts of a surface appear.
- *HDRI (High Dynamic Range Imaging) textures* are used to simulate realistic lighting environments, which we use in this project to simulate realistic backgrounds for our scenes.

By projecting these images onto 3D surfaces, artists and developers can simulate material properties. This trick allows simple shapes to appear more detailed and realistic, significantly reducing the computational cost. In short: more polygons mean more calculations and slower performance, therefore texture maps offer a fast alternative.

What are the challenges of using texture maps? While texture maps are a powerful tool for efficient visual detail, they come with several limitations. The process of UV mapping can be time-consuming and prone to errors such as stretching or misalignment. High-resolution textures, especially when multiple maps are used per material, can consume large amounts of GPU memory. Lastly, textures only simulate surface detail, but they don't physically modify geometry, which means that features like bumps or wrinkles won't cast realistic shadows. In our case, these limitations are negligible: the texture maps are relatively small (compared to the Alembic files), and shadows from geometric details like skin bumps are not the focus of our work. While we initially encountered a texture mismatch between clothing items, this issue has since been resolved, and the clothes now align correctly.

Textures vs realistic clothing. Early video games and animations often used texture maps to represent both skin and clothing directly on the character model. In contrast, modern systems now frequently simulate clothing physically, adding realism at the cost of computational load. For our project we started with realistic clothing simulations based on [11]. However, it's still unclear whether this level of realism actually benefits action recognition models, or if we can achieve similar results with much simpler synthetic data.

How are textures used in this project? The skin color textures are provided by Meshcapade [12] under a Creative Commons Attribution-NonCommercial 4.0 International license. They are grouped into 7 skin color categories based on race as it is done in other papers [11, 13]. The dataset includes 50 textures per assigned sex at birth (i.e., 50 male and 50 female), distributed across the following categories: "african" (10), "asian" (12), "hispanic" (3), "indian" (10), "middle eastern" (3), "south east asian" (5) and "white" (7). For each animation, 7 videos are generated, one for each skin color category. A texture is randomly selected from the available options within each category to allow variation across renderings, and it is applied on the character model, resulting in a realistic synthetic human, see Figure 1.5.

Figure 1.5: Character's texture map vs the rendering of the model with texture applied [12].

1.2.2. Animation, Rigging and Skinning in Computer Graphics

Keyframing. In animation, keyframing is the process of defining important points (key frames) in the timeline of a movement. These key frames mark the most significant positions or poses of an object or character, such as the highest point in a bounce, the moment of impact. Figure 1.6 illustrates this concept with a ball falling and bouncing. By defining these key moments, animators create a structure that can then be filled in with in-between frames (right part of Figure 1.6) to make the movement fluid. This method is foundational not just in traditional animation but also in 3D animation and motion graphics, where digital tools interpolate between key frames to produce smooth transitions. Keyframing allows animators to control the motion of a scene, making it a core technique for bringing static elements to life.

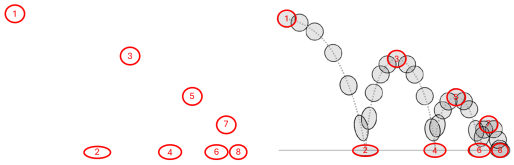
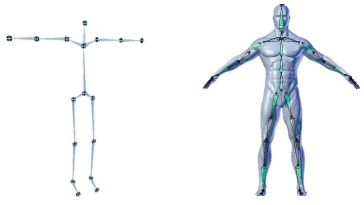


Figure 1.6: Key frames for a ball falling animation. Each number represents a critical moment in the ball's motion. The resulting animation on the right shows interpolation results from the initial key frames and how movement speed can be represented through frames.[3]

Rigging. To animate character models, the animation industry has been using rigs, which are skeletal structures composed of bones and joints Figure 1.7a. Bones define the exact distance between joints, and by rotating joints people can simulate body movements. It is important to note that these rigs have different joints and bones, either depending on the animated character (see human Figure 1.7b vs animal **??**), or based on the level of detail required for an animation (for example a close-up shot of a person's hands requires more precise movements of the fingers, therefore more bones and joints Figure 1.7c).



(a) Human rig composed of bones and joints.[4]

(b) Another rig overlayed on the body model.[5]

(c) Hand rig with more joints for precise control.[6]

Figure 1.7: Examples of rigged and skinned characters from the animation community.

Manual rigging and animation is a tedious process that used to be done manually by graphics artists, where each pose had to be defined in order to create a motion, with interpolation between relevant key

[3] https://sotafoundations2.wordpress.com/2020/02/04/animation-tutorials-day-1-bouncing-ball-2/
[6] https://mocappys.com/how-to-rig-a-character-for-motionbuilder/
[6] https://blenderartists.org/t/rigify-help/532751
[6] https://blenderartists.org/t/rigging-hands-and-thumb/1111225

frames. In order to speed up this process lately people have been recording motions mainly through Motion Capture (MoCap) systems, which have clearly labeled keypoints on a person's body, such that they can be automatically matched to joint positions.

Skinning. Some of the previously shown characters Figure 1.7 are also overlayed over their "skins" or model meshes, but this does not mean they can be properly animated. Meshes are composed of lots of triangles, and by inputting joint positions to a skinning function the points on a mesh decide where to be moved and how the triangle shapes get deformed. Unlike the bones mentioned in rigging, skins are actually flexible just like in real-life. There are different types of skinning functions, but the most simplistic one is Linear Blend Skinning (LBS). After rigging a character, the skeleton is attached to the model mesh, and each vertex is assigned weights based on nearby bones, which get moved by joints.

Equation 1.1 shows the Linear Blend Skinning formula, where each vertex is influenced by a weighted combination of bone transformations. More specifically:

- $\mathbf{v}'$ is the new vertex position (after skinning),
- $\mathbf{v}$ is the original vertex position,
- w_i is the weight of the i-th bone,
- T_i is the transformation matrix of the i-th bone,
- n is the number of bones influencing the vertex.

$$\mathbf{v}' = \sum_{i=1}^{n} w_i \cdot \mathbf{T}_i \cdot \mathbf{v} \tag{1.1}$$

Equation 1.1. Linear Blend Skinning (LBS) computes the new vertex position by blending bone transformations weighted by influence.

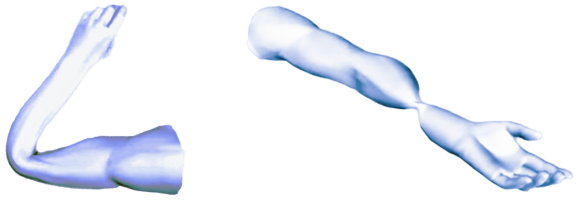


Figure 1.8: Example problems[7] with Linear Blend Skinning (LBS), left: collapsing joints, right: candy wrapper effect.

Linear Blend Skinning (LBS) works fast and is easy to implement, but blending linear transformations (especially rotations) leads to visible artifacts. Some problems of LBS are: (1) collapsing joints, when joints like elbows or knees bend sharply it looks like objects cave in, and (2) the candy wrapper effect, where if bones are rotated too much around their local axis it shows unnatural twists, as can be seen from Figure 1.8. To solve these problems Blend Shapes were introduced.

Blend shapes are a manual animation technique in which artists sculpt specific deformations of a 3D mesh to enable expressive, pose-dependent variations. As shown in Figure 1.9, each variation, such as a smile or yawn, is created as a separate target shape with the same topology as the base mesh (i.e. same number and order of vertices). These target shapes alter only vertex positions, and can be interpolated using scalar weights to produce smooth transitions between expressions, through the formula in Equation 1.2. Blend shapes are typically applied on top of skinning methods like Linear Blend Skinning (LBS) to correct for artifacts or anatomical inaccuracies. Because the deformations are sculpted by hand, they can capture subtle facial motions, as evident in the detailed examples of expressions in the figure. However, this process does not generalize across motions or identities and requires significant manual effort to produce high-quality results.

Variables for interpolating between blend shapes:

- v vertex position in the base mesh
- v_j position of the vertex in the j-th target shape
- w_j is weight of the j-th blend shape,

$$\mathbf{v}' = \mathbf{v} + \sum_{j=1}^{m} w_j \cdot (\mathbf{v}_j - \mathbf{v}) \tag{1.2}$$

Equation 1.2. Blend shapes formula to compute the new vertex position by interpolating pre-defined poses weighted by influence.

Figure 1.9: Example blend shapes for a face, with the base shape in the top left corner and 7 different facial expressions.

To address the limitations of handcrafted corrections, parametric models like SMPL replace blend shapes with learned pose- and shape-dependent deformations, enabling scalable and anatomically consistent animation across a wide range of poses.

[7] https://medium.com/offnote-labs/3d-face-and-body-reconstruction-95f59ada1040

1.2.3. BEDLAM, SMPL and 3D human models

BEDLAM [11] is the framework we used throughout the project to generate videos of synthetic humans performing realistic and consistent movements. Skinned Multi-Person Linear model (SMPL[8]) [10] is a realistic model of the 3D human bodies that simplifies the process of skinning through learned blend shapes and learned joint positions per body type. SMPL is trained on thousands of 3D body scans, enabling it to capture a wide range of human body shapes and natural pose-dependent deformations. Unlike traditional skeletal models that rely only on joint positions, SMPL models the full surface of the body. SMPL model is defined as:

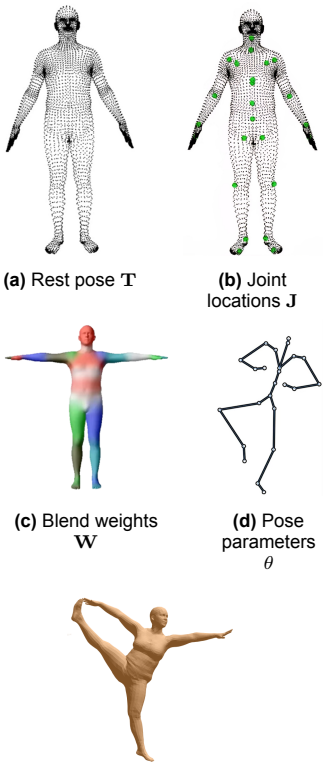
$$T_P(\vec{\theta}, \vec{\beta}) = \mathbf{T} + B_S(\mathbf{S}, \vec{\beta}) + B_P(\mathbf{P}, \vec{\theta}) \tag{1.3}$$

$$M(\vec{\theta}, \vec{\beta}) = \text{LBS}(T_P(\vec{\theta}, \vec{\beta}), \mathbf{J}(\vec{\beta}), \mathbf{W}, \vec{\theta}) \tag{1.4}$$

Equation 1.3. and **Equation 1.4.** The SMPL model takes pose $\vec{\theta}$ and shape $\vec{\beta}$ as inputs, with remaining terms learned from data. This is a simplified definition of SMPL using the Linear Blend Skinning (LBS) function.

SMPL separates body shape (identity β) and joint rotations (pose θ) into distinct parameters, enabling flexible control over both body proportions and movement when generating a human mesh. The inputs to the model are:

- θ: pose parameters controlling joint rotations
- β: shape parameters defining body proportions



(a) Rest pose $\mathbf{T}$ **(b)** Joint locations $\mathbf{J}$ **(c)** Blend weights $\mathbf{W}$ **(d)** Pose parameters θ **(e)** Posed body with SMPL, Eq 1.4.

Figure 1.10: SMPL [10] components.

In addition to these inputs, SMPL relies on several learned components:

- $\mathbf{T}$: average template mesh representing the body in a rest pose. It is combined with the body shape $B_S(\mathbf{S}, \vec{\beta})$ and pose $B_P(\mathbf{P}, \vec{\theta})$, resulting in a subject-specific base mesh.
- $\mathbf{J}$: joint regressor matrix that maps mesh vertices to joint locations, depending on the body shape.
- $\mathbf{S}$: shape blendshape matrix that models identity-dependent shape variations $B_S(\mathbf{S}, \vec{\beta})$
- $\mathbf{P}$: pose blendshape matrix used to produce pose-dependent deformations $B_P(\mathbf{P}, \vec{\theta})$
- $\mathbf{W}$: blend weights matrix defines how smoothly the mesh vertices are rotated around the estimated joint centers, used in the skinning process.

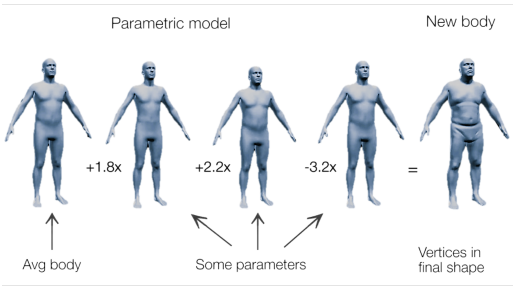


Figure 1.11: Illustration of SMPL's [10] additive body modeling (body math[9]). The average template mesh is combined with multiple learned shape blendshapes, each scaled by a corresponding parameter. These components are linearly added to produce a new, personalized body shape.

Together, these learned components ($\mathbf{T}$, $\mathbf{S}$, $\mathbf{P}$, $\mathbf{W}$, and $\mathbf{J}$) enable the SMPL model to generate realistic human bodies across a wide range of shapes and poses.

How blendshapes work. As illustrated in Figure 1.11, the SMPL model generates new body shapes by linearly combining a mean template mesh with a set of learned shape blend shapes, each scaled by user-defined parameters. This "body math" corresponds to the additive formulation in Equation 1.3. Because SMPL is fully differentiable, it gives animators precise control over body shape and pose by adjusting $\vec{\beta}$ and $\vec{\theta}$.

SMPL uses an additive formulation that separates components like shape and pose, combining them to produce the final body mesh. This modular design makes the model easily extensible, allowing new components to be integrated in a similar additive manner. SMPL-X [14] builds on this by extending the model to include expressive

[7] More info on SMPL website: https://smpl-made-simple.is.tue.mpg.de/
[8] SMPL made Simple: https://www.youtube.com/watch?v=rzpiSYTrRU0

facial features and detailed hand articulation, with a higher-resolution mesh to support these additional details. Because of this additional control, BEDLAM [11] uses SMPL-X to generate realistic movements.

AMASS [15]. Human motion modeling has been challenged by fragmented mocap datasets with inconsistent formats and joint definitions. AMASS addresses this by unifying Motion Capture (mocap) datasets under a shared representation using the SMPL [10] body model. This standardized format enables large-scale analysis and modeling. In our work, we use motion sequences from AMASS due to its animation diversity, and compatibility with SMPL.

1.2.4. Visualizing the results

In this project, data visualization serves as a critical tool for bringing to light the limitations of action recognition models. Visualizations were created during the project with the goal of inviting inspection as well as reporting findings, and they help bridge the gap between meaningful interpretation and raw numerical output. We use visualizations to reveal patterns of bias, variation across conditions camera viewpoint, background, skin color, and specific breaking points, instead of depending only on numerical measures like overall accuracy or loss values, which don't convey the whole story. This reflects the core idea from [17] that "the purpose of visualization is insight, not pictures", highlighting their diagnostic and exploratory value in understanding model behavior.

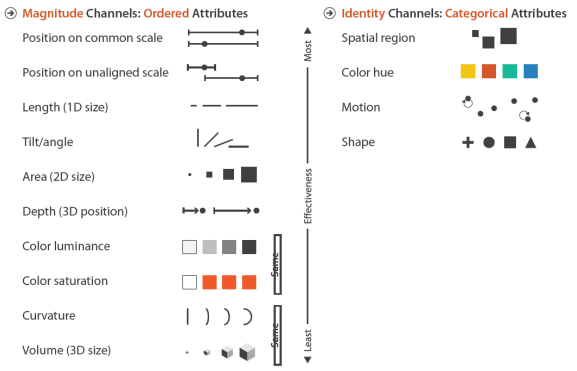


Figure 1.12: The effectiveness of a channel in data encoding depends on the attribute type, from [16]. This ranking guided our selection of channels when designing plots to reveal model biases.

Visualization goals and tasks. The primary aim of our visualizations is to support exploration and interpretation of model behavior across individual variation of visual attributes in the synthetic dataset. These visualizations are designed to address concrete questions about performance, bias, and reliability. Specifically, they help us investigate:

1. Which actions perform well or poorly under which conditions?
2. How do different settings for camera viewpoint and background affect the similarity between the synthetic data and the training data?
3. How do changes in visual attributes such as skin color affect misclassifications?
4. Are differences in performance across skin tones statistically significant?

The underlying data consists of model performance metrics (action recognition model accuracy) evaluated across multiple categorical conditions: skin color (7 categories), camera viewpoints (near, far), backgrounds (autumn park, konzerthaus, stadium), and various action labels. This information is structured as tabular data, with each row representing the performance of a specific model under a given combination of attributes. We control for confounding by altering only one variable at a time, ensuring interpretability of observed differences.

Following the task taxonomy described in [16], our analysis goals map onto three core types of tasks:

- Discover: Identify unexpected performance drops, patterns, or outliers, such as an action that fails only in a specific background or a skin tone that triggers frequent misclassification.
- Compare: Assess differences between models, attribute settings (e.g., best vs worst performing background), and performance across skin tones or camera angles.
- Summarize: Observe aggregate patterns such as overall model robustness or performance distribution across categories.

These goals and tasks guide the design and interpretation of our plots, enabling nuanced inspection of the impact of single attribute changes on the accuracy of action recognition models.

1.2.5. Design choices and best practices in our visualizations

To ensure our visualizations support insight rather than just displaying information, we followed best practices in data visualization and design, from [16]. We selected the visual channels that were most suitable for the data types, based on their ranked effectiveness in question, Figure 1.12. Based on this ranking, we selected more effective channels for our visualizations, see Figure 1.13. For quantitative data (e.g. accuracy percentage of action recognition models), we used position along a common axis, length, and bar channels known to support precise magnitude comparison. For categorical data such as action labels, camera viewpoints, or background types, we employed spatial separation through grouping, and color, which help convey identity distinctions without implying order. For example, bar charts were chosen to show accuracy variation across camera viewpoints and actions, preserving both categorical clarity and quantitative precision. We also used color consistently to highlight individual conditions across actions while supporting accessibility through readable palettes, and colorblind symbols marks. For ease of comparison, we decided to sort the categorical attributes based on the value of their magnitude. Overall, these design choices aim to aid the reader to explore models in more depth, helping uncover patterns of bias and enabling more informed interpretation than metrics alone allow.

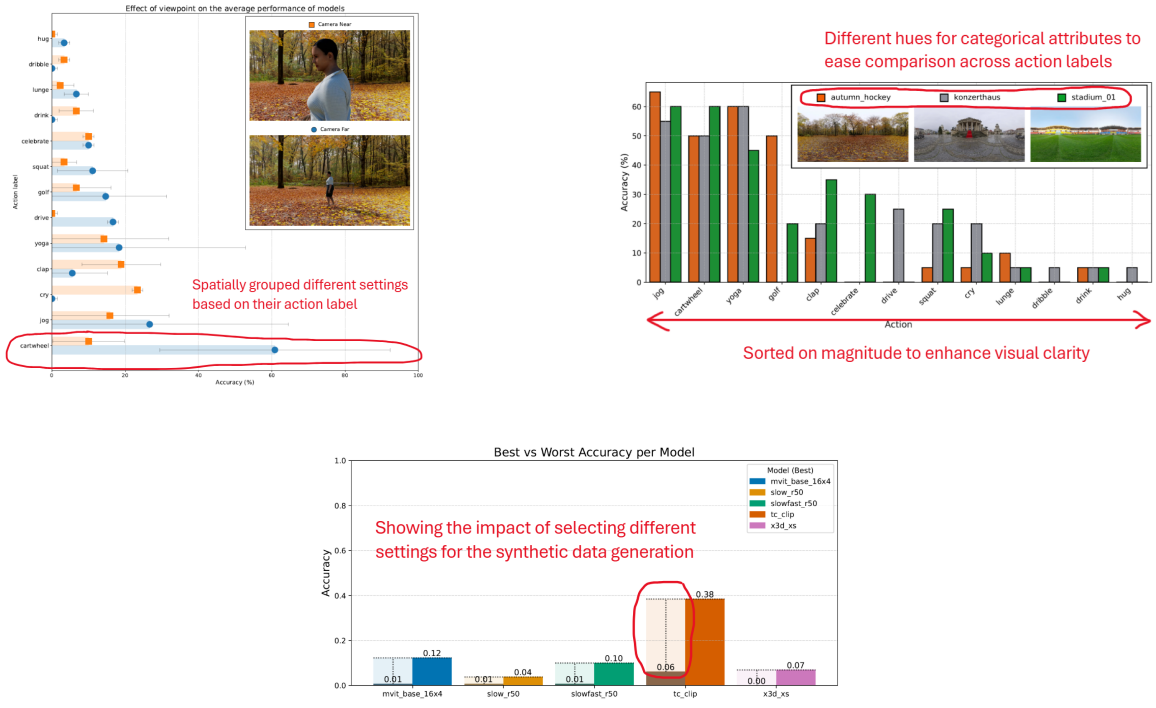


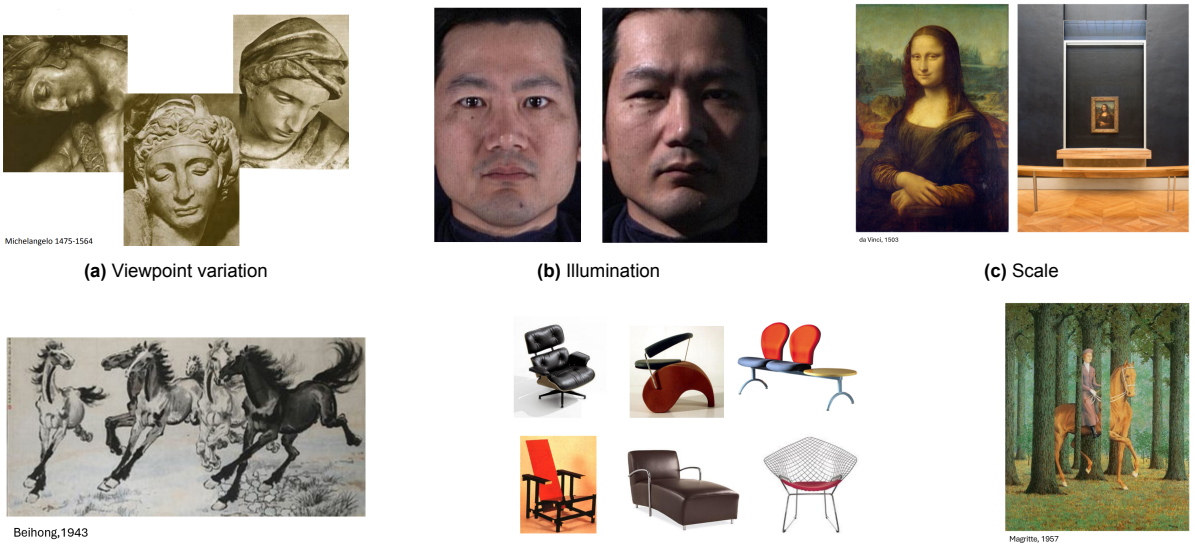
Figure 1.13: Design decisions applied in our visualizations to enhance interpretability. We used spatial grouping for ordering the main categories (action labels or models), and color hue for secondary categorical distinctions like background or camera viewpoint. Sorting by magnitude allowed for easier comparison across conditions. These choices enable readers to detect performance patterns and potential biases more effectively.

1.3. Intro to Computer Vision

At its core, Computer Vision seeks to replicate aspects of human visual perception, allowing computers to process visual input and perform tasks such as recognizing what objects are in an image, detecting where something is located, and understanding motions. Although these processes may feel effortless to us, they pose significant challenges when implemented by computers, since the tasks differ substantially depending on the nature of the visual input and the desired output. The task we are focused on is Action Recognition, which involves identifying what action is taking place in a video (e.g., walking, running, dancing). Unlike static image tasks, action recoognition requires understanding how motion unfolds over time.

Why is Computer vision so hard? Real-world situations contain a wide range of variations in visual input, including [18]: changes in viewpoint, illumination, scale, deformation, occlusion, and intra-class variation, Figure 1.14. In the context of action recognition these challenges also apply:

- *viewpoint* variations can make actions appear different depending on the angle of observation, e.g., a movement might be clear from a frontal view but ambiguous from the side.
- *illumination* affects how the scene is perceived, changes in lighting conditions, such as indoor or outdoor environments or different times of day, can alter the appearance of actions.
- *scale* influences how visible an action is, movements performed too far from the camera or with subtle gestures may be difficult to detect, while actions too close may be partially out of frame.
- *deformation* refers to how the human body naturally stretches and bends during movement, which can complicate recognition across frames.
- *intra-class variation*, means that actors can look differently and perform the same action, or the same action looks different when performed by different people, and sometimes even if the same person performs the same thing, it might look different
- *occlussion* occurs when key parts of an action are hidden, either by the actor's own body, other objects, or the scene boundaries.



(a) Viewpoint variation **(b)** Illumination **(c)** Scale

(d) Deformation **(e)** Intra-class variation **(f)** Occlusion

Figure 1.14: Common challenges in Computer Vision [18]. Shows the differences that impact model's understanding of a scene.

On top of that, the tasks are not as clearly defined as we think. Unlike humans, computers lack the flexibility to reinterpret or adapt a task when faced with new situations. In the field of action recognition, the tasks differ in the types of actions being performed, the camera perspective (first-person or third-person view), the number of people in a scene or the amount of concurrent actions performed by a person at once. Some papers even define break down actions into smaller sub-actions [19]. This diversity in task definition complicates both model development and evaluation, making it difficult to fairly compare approaches or to generalize results across models or datasets.

1.3.1. Existing Action Recognition datasets

In this project we test models, but to an extent we also test the datasets that they were trained on. In Action Recognition the training data has a major impact on a model's ability to generalize to unseen situations. This is particularly important for models trained on monocular RGB video, as they often rely heavily on visual cues such as background, clothing, or lighting conditions to recognize actions [20, 21, 22]. Therefore, to get comparable results, we test only models pretrained on the same dataset. In our case, the models are pretrained on Kinetics-400 [23], a large-scale dataset which includes everyday human actions, has pretrained weights publicly available[10] and is being widely used [4, 9, 24, 5]. The scale and diversity of Kinetics-400 make it a practical choice for analysis, in this project being used to understand how visual information correlates with specific actions. Other commonly used action recognition datasets include:

- UCF-101 [25]: a relatively small dataset spanning 101 action categories, featuring sports and human-object interactions, captured from YouTube in third-person perspective.
- HMDB51 [26]: similarly small dataset with 51 action classes, filmed in third person view, generally used for evaluation of models and not training.
- AVA [27]: a more complex dataset third person, consists of scenes from movies with actions localized spatially per frame, with multiple people potentially having multiple labels.
- Something-Something V2 (SSv2) [28]: a first-person dataset that emphasizes interactions with objects rather than full-body human motions. As a result, the set of actions represented differs significantly from those found in the previously mentioned third-person datasets.
- EPIC-Kitchens-100 [29]: a large-scale dataset focused on kitchen activities, with limited action diversity, background variation, and lighting conditions.
- Charades [30]: a dataset where participants acted out sentences composed of objects and actions from a fixed vocabulary (like a game of Charades), resulting in somewhat controlled scenarios.

These datasets vary in size, viewpoint (third-person vs. first-person), granularity and amount of actors per scene. In our setup, we focus on third-person datasets similar in style to Kinetics-400, such as UCF-101, HMDB51, AVA or Charades. In contrast, we do not expect our current method to generalize well to first-person datasets like SSv2 or EPIC-Kitchens-100 without further adaptation.

Prior work has explored how to reduce the reliance of models on visual context, either by creating appearance-free models or by constructing datasets that explicitly control for visual cues [20, 31]. While we see this as a valuable direction for future development, there remains a need to evaluate the models that are already in use. As the EU AI Act [1] starts being enforced, the need for tools that help audit and understand model behavior is becoming more urgent. In this work, we aim to contribute a step in that direction by developing a method for numerically evaluating the extent to which action recognition models rely on visual features.

1.3.2. Action Recognition models

In this section, we provide a brief overview of the action recognition models evaluated in this study and discuss their relevance as baselines for our analysis.

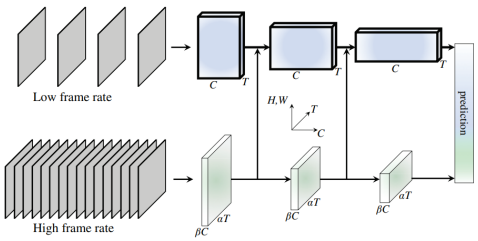


Figure 1.15: SlowFast network [4], with a Slow pathway (top) for spatial semantics and a Fast pathway (bottom) for fine motion, enabling joint modeling of scene context and temporal dynamics.

The **Slow** [4] network serves as a foundational architecture for video action recognition by focusing on semantic understanding through sparse frame sampling. It processes a video clip using a large temporal stride to capture only one out of (typically) 16 frames. This design enables the model to focus on scene semantics such as objects, background context, and sustained body poses (e.g. actions like yoga), instead of fast motions. The Slow pathway is instantiated as a 3D convolutional neural network and it operates on low temporal resolution inputs, allowing spatial reasoning over extended durations. This approach assumes that many important semantic cues in video are relatively stable over time, which holds true for action recognition datasets recorded from fixed camera viewpoints and is applicable to our project. Although the Slow pathway alone is capable of modeling persistent visual features, it lacks sensitivity to rapid motions. Therefore, in [4] it has primarily served as a baseline for more temporally sensitive architectures, such as the SlowFast network.

SlowFast [4] is a biologically inspired model that separates the processing of slow-changing semantics and fast motion using two distinct pathways. The Slow pathway captures spatial semantics at a low frame rate, while the

[10] Gluon models pretrained on Kinetics-400: https://cv.gluon.ai/model_zoo/action_recognition.html?#kinetics400-dataset

Fast pathway processes frames at a higher rate to focus on rapid motion. This two-stream processing design draws inspiration from the P-cell (sensitive to fine spatial detail and color) and M-cell (tuned to fast motion) pathways in the human visual system. This approach has led to state-of-the-art performance on benchmarks like Kinetics-400 [23], Charades [30], and AVA [27].We include SlowFast in our evaluation not only for its conceptual novelty, but also because it has consistently set strong baselines on major action recognition benchmarks[11].

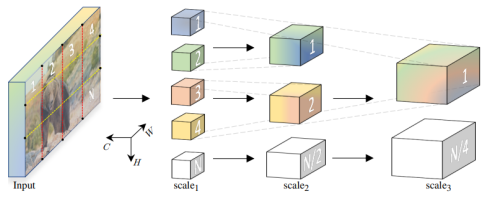


Figure 1.16: MViT [9] builds a hierarchical representation by reducing spatial resolution and increasing channel capacity, capturing features at multiple levels of granularity.

MViT [9] (Multiscale Vision Transformer) is a transformer-based architecture designed for efficient visual representation learning from both images and videos. MViT introduces a hierarchical design with multiple scale stages. These stages progressively reduce spatial resolution while increasing feature dimensionality, forming a multiscale feature pyramid that captures both fine-grained and high-level information. MViT is notable for training entirely from scratch, without reliance on large-scale external datasets like ImageNet [32] and still achieving state-of-the-art results on benchmarks such as Kinetics [23] and AVA [27]. MViT models show understanding of temporal cues without, avoiding spatial biases, which is a potential pitfall for such models. With implications for real-world AI systems in fields like robotics, surveillance, and autonomous navigation, MViT is a promising step toward more effective and reliable video interpretation, making it a strong choice for our analysis.

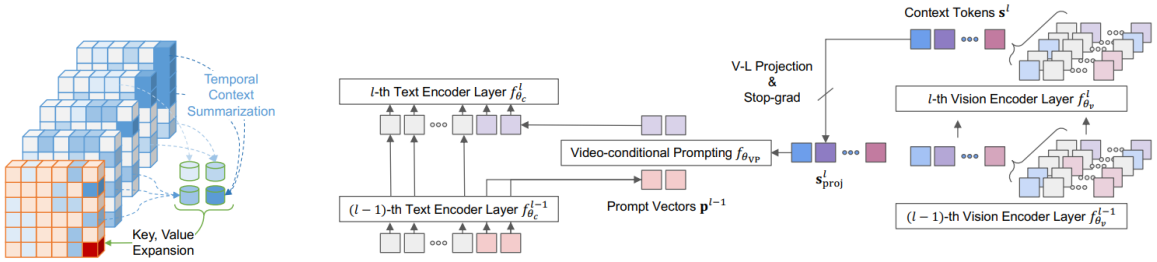


Figure 1.17: The TC-CLIP [8] architecture combines two components: on the left, the Temporal Contextualization (TC) module selects informative patch tokens across video frames and summarizes them into context tokens for global temporal reasoning, while on the right, the Video-conditional Prompting (VP) module uses these tokens to condition text prompts, producing video-specific textual representations.

CLIP (Contrastive Language–Image Pre-training) [33] is a multimodal model trained on image-text pairs. Instead of learning to match images to fixed labels, CLIP learns to align image features and language features. This allows it to perform zero-shot classification, recognizing new categories just by being given their names as text prompts, which makes it adaptable to different visual tasks without additional training. While CLIP is effective for image labeling, it lacks mechanisms for modeling motion or temporal structure. **TC-CLIP** [8] extends CLIP to video by introducing Temporal Contextualization (TC), which summarizes visual changes across frames into context tokens, capturing global action cues. A Video-conditional Prompting (VP) module then uses these tokens to adapt text prompts for each video instance. TC-CLIP is well-suited for our project as it supports zero-shot video understanding using only natural language, without additional labels or retraining. Its video-specific prompts act as an implicit labeling mechanism, which may introduce or amplify bias, making it a strong candidate for studying misalignment between visual and linguistic representations.

X3D [5] is a family of efficient video recognition models that progressively expand a minimal 2D image classification network along axes such as temporal duration, frame rate, spatial resolution, width, bottleneck width, and depth (see Figure 1.18). X3D starts with a 2D base model and performs stepwise expansions, evaluating one axis at a time for its impact on accuracy and computational cost. This approach resembles a greedy coordinate descent in the hyperparameter space and allows X3D to reach competitive accuracy while maintaining low computational requirements. Its lightweight design is ideal for inference-heavy applications like ours, and it is trained from scratch, avoiding biases from image-based pretraining such as ImageNet [32].

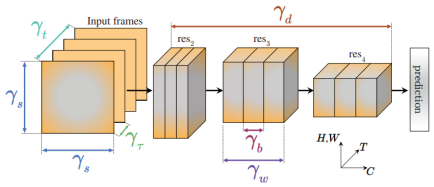


Figure 1.18: X3D network showing how it expands a base 2D network along multiple axes $(\gamma_S, \gamma_t, \gamma_\tau, \gamma_b, \gamma_w, \gamma_d)$.

[11] Slowfast performance on benchmarks: https://paperswithcode.com/paper/slowfast-networks-for-video-recognition

1.4. Intro to EU AI Act

The European Union's Artificial Intelligence Act (EU AI Act) [1, 2] is the European Union's first harmonized legal framework that specifically addresses the development and deployment of AI systems within the Union. The Act represents a significant step towards ensuring that AI technologies are developed and used responsibly within the European Union. It aims to guarantee that AI systems are safe, ethical, transparent, and trustworthy, while maintaining a strong commitment to protecting fundamental rights and values from the Charter of Fundamental Rights [34]. By doing this, the Act seeks to strike a careful balance between encouraging innovation and preserving the core principles of democratic societies.

Similarly to the EU's General Data Protection Regulation (GDPR) [35], the AI Act has extraterritorial reach. This means that AI providers outside the European Union must comply with the Act if their systems are used within the EU market [36]. As researchers contributing to global advancements in AI, we have a responsibility to engage with the AI Act and respect such regulatory efforts, regardless of where we are based.

How do regulations differ per application type? The Act is divided into acceptable and unacceptable risks, where the latter consists of AI systems that are prohibited being used in the EU market. The acceptable risks are further divided into: high risk, limited risk and minimal risk, depending on the impact of the application, Figure 1.19. The EU AI act imposes stricter requirements on applications with greater potential for harm to health, safety and fundamental rights, particularly in sensitive areas such as law enforcement, healthcare, education, and employment. One of the Act's defining features is its regulation of general-purpose AI (GPAI) systems, such as large language models, or models that can be adapted for various use cases, which must meet transparency obligations and adhere to copyright laws. This is particularly relevant to our field, since in Computer Vision researchers often develop foundational models (in the Act referred to as GPAI) that are later fine-tuned or deployed by companies for their specific applications. These backbones, if not carefully assessed, can create or amplify biases, making it essential to consider their downstream impact before release.

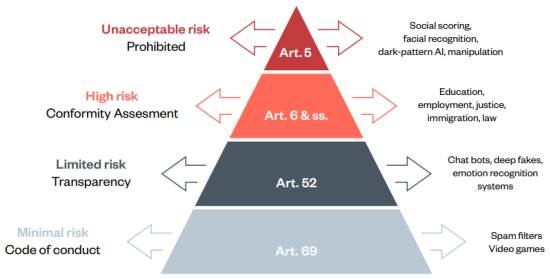


Figure 1.19: EU AI Act risk-based approach [2]. At the top, Unacceptable risk systems are prohibited. High-risk systems require strict compliance and assessment. Limited-risk applications must meet transparency requirements. Minimal-risk systems follow voluntary codes of conduct. Our focus is on analyzing action recognition systems, which can be placed in different risk categories depending on the application.

What type of risk are we focusing on? Depending on the downstream task, Human Action Recognition models could fit in all the risk categories. We are expecting that HAR models that fall under "unacceptable risk" category will not be deployed, so we do not concern our analysis with that. Instead, in this project we focus on High-Risk AI Systems (HRAIS), which represent the most critical level at which AI models can be legally deployed under the EU AI Act. This category includes systems that may significantly affect individuals' health, safety, or fundamental rights. We specifically examine Action Recognition models, as their deployment in areas like surveillance, security, or human-computer interaction, can impact the decision making process and thus falls within the scope of high-risk applications defined in Annex III[12] of the Act. Even though our focus is primarily on HRAIS, the methodology can also be applied to evaluate models in the Limited risk category.

How do we increase transparency and explainability? Transparency is a core principle of the EU AI Act, particularly for HRAIS. Article 13[13] mandates that providers (i.e. developers) are required to ensure their systems are understandable and interpretable by users. This includes providing clear documentation about the system's purpose, capabilities, limitations, and the role of human oversight (e.g. explicitly mentioning how and why they are monitoring the application). In this project, we focus specifically on transparency and explainability as measurable properties, aiming to support researchers and developers in assessing their models quantitatively. While the EU AI Act places the burden of compliance primarily on providers, it does not currently offer standardized tools or methods for evaluating these requirements in practice. The EU AI Act currently requires a conformity assessment before releasing a high-risk model, which can be fulfilled either through self-assessment or by engaging with a third party. Decisions are not grounded in empirical evidence and the Act lacks consistency and comparability across models [37]. Rather than relying solely on static documentation and descriptions, introducing quantifiable metrics could allow for more objective comparisons between models. At the moment, there is a disconnect between political guidelines and the development of technical standards, creating an implementation challenge for both regulators

[12] Annex III: https://artificialintelligenceact.eu/annex/3/
[13] Article 13: https://artificialintelligenceact.eu/article/13/

and developers seeking compliance. The current required technical documentation, explained in Annex IV[14] is too vague on the technical side. By exploring numerical ways to assess transparency and explainability, this project contributes to more trustworthy AI development and facilitates alignment with the EU AI Act's expectations.

Why are we so scared about AI? AI systems seem unpredictable to programmers and researchers, particularly when contrasted with the determinism of traditional methods. We tend to trust systems whose outputs we can predict and explain, while AI introduces uncertainty since it may produce different answers for the same input, and the reasoning behind its responses is opaque. This becomes even more concerning when combined with the overconfidence of models. This mismatch between apparent confidence and actual reliability makes it harder to assess whether we can trust them. On top of that, these models are compelling enough that many users blindly accept their output. This is especially dangerous in high-stakes settings, when AI is used in decisions that affect people's lives, e.g. surgery or job applications [38]. Because the models are made up of building blocks with little regard to explainability or meaningful structure, it's difficult to understand their outputs. We are scared because AI systems are powerful and useful, but lack the transparency and accountability expected from technologies with such influence.

1.5. Intuition behind our approach

The main idea of this project is that we are trying to evaluate a task with a lot of variation, Human Action Recognition, by converting it into a controlled experiment. In real-world experiments, results are often influenced by uncontrollable elements known as *confounding factors* (Figure 1.20), such as differences in lighting, environment, or subtle variations in how a person moves. Besides these confounders, experiments also involve *independent variables*, which we intentionally manipulate, and *dependent variables*, which are the outputs of those manipulations. Our goal is to minimize the impact of confounding factors to allow for reliable, measurable comparisons.

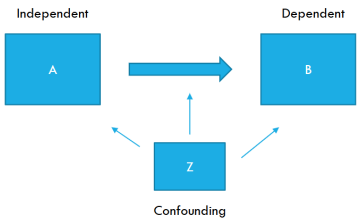


Figure 1.20: Independent variables, confounding factors, and dependent variables, showing how they are related in an experiment[15].

Take the example of recording multiple individuals performing the same action, discrepancies inevitably emerge, from how they move, to the clothes they wear, to differences in camera positioning or ambient light. Some of the difference can be minimized by controlling the recording conditions, for example by filming indoors with constant lighting. But motion variability is difficult to eliminate, since people cannot exactly replicate the same movement, even if it is the same person performing it. Therefore we believe using synthetic data, with exactly the same movement, leads to measurable and comparable results. This allows us to reduce variability and focus exclusively on how changes in independent variables affect the outcome. By doing so, we can better isolate the effect of specific factors and gain clearer insights into the behavior of HAR models.

[14] Annex IV: https://artificialintelligenceact.eu/annex/4/
[15] Image taken from the CSE3500 Human Computer Interaction course from TU Delft.

2

Scientific paper

Figure 1: Qualitative analysis showcasing potential racial bias in action recognition models. Predicted labels per video at the bottom right of the frame. In Video 1 (top row), all sequences depict the same motion performed in the same environment, with only the actor's skin tone changed. The model correctly labels the lighter-skinned person as “cartwheeling”, while misclassifying the darker-skinned person as performing “capoeira” an Afro-Brazilian martial art that blends dance and acrobatics. In Video 2 (bottom row), similar inconsistencies occur for the action “jumpstyle dancing” depending on the perceived race of the actor. See: https://youtu.be/amygAq-Sqc4. This suggests that some action predictions rely on visual attributes rather than on the movement.

Abstract

Human Action Recognition (HAR) models are increasingly deployed in high-stakes environments, yet their fairness across different human appearances has not been analyzed. We introduce a framework for auditing bias in HAR models using synthetic video data, generated with full control over visual identity attributes such as skin color. Unlike prior work that focuses on static images or pose estimation, our approach preserves temporal consistency, allowing us to isolate and test how changes to a single attribute affect model predictions. Through controlled interventions using the BEDLAM simulation platform, we show whether some popular HAR models exhibit statistically significant biases on the skin color even when the motion remains identical. Our results highlight how models may encode unwanted visual associations, and we provide evidence of systematic errors across groups. This work contributes a framework for auditing HAR models and supports the development of more transparent, accountable systems in light of upcoming regulatory standards.

Keywords: Human Action Recognition, Synthetic dataset, Bias analysis.

1 Introduction

Can an action recognition model mistake a cartwheel for capoeira simply because the actor has a darker skin tone? Such questions are central to evaluating fairness in Human Action Recognition (HAR), where predictions may reflect underlying bias, see Figure 1. We present a framework for auditing fairness in Human Action Recognition (HAR) models using synthetic data. By systematically varying visual attributes our approach exposes potential biases in model predictions that may otherwise remain hidden in real-world datasets. This is increasingly relevant as HAR models are integrated into high-stakes domains such as security, autonomous driving, healthcare [1], and regulatory frameworks like the EU AI Act [2, 3] demand greater transparency and accountability from developers of AI systems. With this paper we are trying to assist policy makers, law enforcers and AI engineers in developing more robust and fair systems, making it easier to verify their compliance with emerging legal and ethical standards. To this end, we developed a framework for generating synthetic video datasets with complete control over a subset of visual attributes. Modifying a single attribute allows us to isolate their effects and measure bias without introducing confounding factors.

HAR models exhibit bias because they do not always learn to recognize actions based purely on human movement. Instead, they may rely on visual attributes that are not directly related to the action itself, such as clothing, background, or body appearance [4, 5, 6, 7, 8]. For instance, a model might associate a Hawaiian shirt with drinking a cocktail, even though the shirt has nothing to do with the action being performed. This suggests that the model could be picking up on superficial visual patterns in the training data, rather than truly understanding the motion, which raises concerns about fairness and reliability.

In this project, we identify how sensitive models are to changing ethical attributes. Regulatory frameworks like the EU AI Act [2, 3] highlight risks such as opacity and data dependency, which can lead to discriminatory outcomes in AI systems. Motivated by these concerns, we focus on potential biases linked to race or skin color, which may undermine fundamental rights like non-discrimination [9]. Our analysis does not include general action recognition tasks like industrial processes, or animal monitoring, instead, due to the inherently human nature of our selected attributes, we

limit the analysis to the HAR domain.

Some approaches for evaluating biases have been to manually annotate [10], or derive information from existing labels [11], and then assess model performance for each group. An emerging trend has been to generate controlled synthetic images using neural networks, for facial recognition [12], or human pose and shape estimation [13]. Similarly, for the task of Human Action Recognition we require video data in which specific attributes can be controlled. While prior work such as STAGE [13] has demonstrated bias analysis in individual frames, HAR introduces an additional challenge: motion across frames. This requirement makes extending frame-level approaches like STAGE to the video domain particularly challenging and prone to generation defects, resulting in less control.

BEDLAM [14] allows us to generate physically plausible and temporally consistent videos while programmatically controlling visual attributes. To avoid introducing confounding factors because of potential artifacts, we adopt the fully simulated approach through BEDLAM. The simulated approach enables isolated interventions such as changing skin tone, so we can systematically evaluate how these attributes influence model predictions.

This paper aims to investigate biases from human action recognition (HAR) models in a simulated environment. Specifically, it addresses the main research question: ***How sensitive are Human Action Recognition (HAR) models to single visual attribute changes in synthetically generated RGB videos?*** To explore this question, the paper focuses on two sub-research questions:

SubQ1. How well do HAR models trained on real data generalize to synthetic videos?

SubQ2. How can we measure significant differences in HAR predictions when changing personal characteristics?

To address the research questions, we present the following contributions: we evaluated publicly available action recognition models directly on fully synthetic data without fine-tuning, observing that while some models under-perform, others yield promising results. Finally, we focused on skin color as a key parameter to investigate model bias, identifying cases where predictions shift significantly with changes in appearance.

2 Related work

Past research has been exploring biases across various domains of deep learning. This includes studies on gender bias in text-to-image generative models [15], and visual bias in vision-language models [16]. Within the field of action recognition, prior research has investigated model behavior under occlusion [17], as well as their robustness to a wide range of noise types [18]. Our paper focuses on making a framework for auditing action recognition models and the datasets they are trained on. We want to identify if they have any inherent biases towards ethical attributes such as skin color. Previously there have been other papers who made similar auditing, testing and diagnostics frameworks [13, 19, 20]. STAGE [13] focuses on the task of Human Pose and Shape estimation (HPS) models, while [18] focuses on robustness analysis on 90 perturbations (e.g. noise, blur, camera movement). At their core these tasks are similar to action recognition, but operate on individual frames, while action recognition requires reasoning over entire motion sequence.

2.1 What makes HAR so challenging?

Human Action Recognition (HAR) using RGB monocular data is a complex task due to the large variety of conditions under which actions occur. Actions are frequently associated with specific environments, camera angles, lighting conditions, and backgrounds [21]. This association is likely due to their **inter-class variability**, since videos of actions under the same label occur in similar settings. For instance, actions like "playing football" typically take place in outdoor fields under natural lighting, whereas "cooking" is performed in a crowded indoors space with artificial lighting. Actions appear different from varying viewpoints [22], so many models assume a fixed viewpoint [23, 24]. This strong correlation between actions and their typical visual contexts can lead models to overfit on these appearance cues, reducing their generalizability across diverse scenarios.

We identified **viewpoint** and **background** as important attributes that do not have an obvious default value, and that have a major impact on performance when generating synthetic data. Previous works have explored the effects of changing the camera angle [25], as well as adding a loss metric for the background [7], but they both use synthetic data for training the models instead of testing. In our own experiments, we also observed that the viewpoint and background changes significantly impact model performance, which is why we chose to control these factors manually, and we limit their impact through ablation studies.

Human Action Recognition (HAR) faces significant challenges due to **intra-class variability** [26, 27, 28]. This variability arises because individuals perform the same action differently: variations in speed, style, and body movements are common. Even the same person may not fully replicate an action across multiple instances [28]. The lack of replicability introduces unwanted confounding factors, which affect measurements of independent variables. This makes it difficult to conduct meaningful analyses of attribute changes in real (recorded, non-synthetic) datasets. Therefore, our study leverages synthetic data to systematically examine these specific variations under controlled conditions, by replicating the exact motion over multiple videos.

2.2 GenAI vs Recorded vs Simulated benchmarks

With the rise of video generation models like Veo 3 [29], as well as no-longer-state-of-the-art approaches like SORA [30] and other recent methods [31, 32], you can't help but wonder if such tools could be used to generate video data. Although producing realistic videos has traditionally been challenging, due to issues with temporal coherence and the introduction of visual artifacts, Veo 3 demonstrates how rapidly these models are evolving, showing noticeably fewer glitches than earlier approaches. This progress suggests that we are approaching a point where generating visually coherent videos may become

feasible. However, our goal requires generating multiple videos that differ by an individual attribute, and this level of controlled consistency is unlikely to be achievable with diffusion-based models, given their potential for introducing artifacts. Therefore, we will not be using generative AI for creating our data.

Another way of auditing would be through recorded videos of people with different visual attributes performing the same action. A downside of the recorded approach is that human actors introduce uncontrollable variation [33] (e.g. timing, joint positioning, or scene lighting), which constitute confounding factors. By contrast, synthetic data ensures perfect reproducibility, allowing us to pinpoint model behavior under precisely defined conditions. This also makes the influence of the independent variable that we alter (skin tone) measurable. Furthermore, synthetic environments are highly scalable and can be adapted retroactively (e.g., changing camera angles or background) without requiring entirely new data collection efforts, thus we prefer the simulated approach.

We generate our synthetic dataset through a simulated approach, using BEDLAM [14]. A recent trend in Computer Vision has been using simulations to generate a large amount of high quality training data [34, 35, 36, 37, 38]. This approach has proven to be highly effective for training models, that are then tested on real data. The good performance has made us question whether the inverse method is possible. Hence, after training the models on real data, we are testing them on synthetic data.

2.3 The golden goose: BEDLAM

BEDLAM [14] showed that training human-pose-and-shape estimation models entirely on synthetic images can still achieve state-of-the-art performance on real data. Their method makes sense for their work because adopting the simulated route made the annotation process simpler, whereas annotating full bodies is a tedious and error-prone process. Although label construction is much easier in our action-recognition study, we use BEDLAM's robust simulation pipeline for a different objective: we begin with models that have been pretrained on real videos and evaluate their resilience on a synthetic benchmark. In other words, where BEDLAM relied on synthetic data for training, we reverse the setting to a controlled, scalable evaluation environment for real-world models. We use the BEDLAM rendering framework[1] to generate our synthetic videos. BEDLAM makes the generation of the dataset relatively simple, since we do not have to worry about the variability of body models, clothing and animations. Without BEDLAM, achieving realism would have been difficult.

SMPL [39] is a model for representing realisting 3D human body meshes. Given joint positions along with an input mesh, SMPL can predict vertex positions of the skinned characters with the help of learned blend shapes. SMPL has been widely used in recent papers [13, 25, 40, 41, 42] related to synthetic human data because it is more accurate than previous human body models, and compatible with rendering software like Unreal Engine. SMPL-X [43] is a more expressive version of SMPL, in terms of hands and face movements. This expressivity improvement is relevant for us when generating actions such as "drinking" since they rely on more precise movements of the body which would not have been possible with just SMPL. BEDLAM uses SMPL-X to generate realistic 3D body movement.

BEDLAM uses animations from AMASS [44] to generate realistic videos of synthetic humans. AMASS is a large-scale collection that merges multiple motion-capture datasets under the unified SMPL model. Conveniently for us, each AMASS sequence comes with an associated action label. Even though AMASS is a large dataset, the scope of action recognition models is broad, and, while good for human pose estimation tasks, the limited amount of motions could constrain future work for action recognition. We avoid this limitation by generating videos only for action labels that match semantically in both AMASS and the dataset the models are trained on.

2.4 Evaluated HAR models

In this paper we test models, but to an extent we also test the datasets that they were trained on. Models trained on monocular RGB video data tend to learn visual cues [4, 5, 7]. Therefore, to get comparable results, we test models pretrained on the same dataset. Kinetics-400 [45] is a dataset that has a large amount of pretrained action recognition model weights [46, 47, 48, 49] available[2]. Because of this and the scale of Kinetics-400, for our experiments, we evaluated models pretrained on it.

We evaluated open-source foundation models, which makes them likely to be used in real-life applications. We did not re-train these models, instead we took models pre-trained on the Kinetics-400 dataset. This choice keeps our method future-proof and gives "deployers" [3] a practical way to check their models before release. Each model had to perform single-label human-action recognition, and we needed at least a rough understanding of the training data to create representative synthetic videos. Using this criterion, we analyzed five models, out of which only three achieved sufficient accuracy on the synthetic data to make subsequent bias analysis worthwhile, as low-performing models would reduce the insight gained from the bias tests because of near-random guessing.

MViT [47] combines the idea of features pyramids with vision transformers, enabling the model to recognize actions at varying scales. MViT is well-suited for analysis because it can recognize both large-scale movements like cartwheeling and small-scale actions like crying.

SlowFast [46] couples a low-frame-rate pathway that learns detailed spatial semantics with a high-frame-rate pathway that focuses on subtle, rapid motion. Fusing the two paths enables understanding across temporal scales, from sustained actions like yoga, to rapid movements like a lunge.

[1] BEDLAM render repository:https://github.com/PerceivingSystems/bedlam_render

[2] Gluon models pretrained on Kinetics-400: https://cv.gluon.ai/model_zoo/action_recognition.html?#kinetics400-dataset

TC-CLIP [48] compresses the most informative patches in each clip into a few Temporal Context (TC) tokens. It then feeds these tokens to the Video-Conditional Prompt (VP) module, giving both vision and language branches a shared clip-level memory that excels on long or partly off-screen actions such as the jog videos in our synthetic dataset.

We also tested **X3D** [49] and **Slow** [46], but their near-random performance on the baseline synthetic dataset led us to drop them from the bias analysis.

3 Methodology: Controlled Bias Auditing

This study presents a proof-of-concept methodology for auditing bias in Human Action Recognition (HAR) systems using synthetic video data. We aim to evaluate whether publicly released HAR models behave robustly when exposed to controlled variations in monocular RGB video inputs, as might occur in real-world settings [45, 50, 51, 52, 53, 54, 55]. With the setup explained in this section, we aim to test whether synthetic data can serve as a reliable tool for isolating and evaluating potential biases in model predictions.

Framing the problem as a controlled intervention, our approach enables systematic bias evaluation in HAR systems with minimal confounding factors. In order to validate the use of synthetic data as a viable testing ground, we first evaluate whether action recognition models can produce correct predictions on fully synthetic video data. To ensure that any observed differences in model predictions stem solely from the manipulated attribute, we fix all other variables: the action performed, the environment, the camera position, the clothing, and all other visual attributes related to the actor's features, while one specific attribute (skin tone) is varied. If a model changes its prediction when only the actor's skin tone changes, we flag this as potential evidence of biased behavior. This setup enables us to isolate and evaluate the influence of individual attributes on model predictions, laying the foundation for a replicable bias auditing framework.

3.1 Key design principles

Human Action Recognition (HAR) is a field that encompasses a large variety of actions categories and we cannot realistically check all possible action types. Our focus was to generate data similar to the Kinetics-400 [45] dataset, but it is entirely possible that a similar approach can be applied to other datasets. Therefore we describe a framework that can be used to generate similar data for other use cases.

To evaluate models without fine-tuning, we generate synthetic data that is similar to Kinetics-400, while remaining fully controllable and replicable. Real-world datasets contain too much uncontrolled variation, and current generative methods introduce artifacts that interfere with controlled experiments. Instead, we simulate realistic videos where only specific variables are changed. To keep the analysis interpretable, we constrain the dataset along three dimensions: types of actions, actor attributes, and scene appearance. This design enables targeted evaluations without overwhelming complexity.

Action constraints. We generate a controlled dataset where each video features a single actor performing a single action. This setup is meant to be the simplest way forward, since it is compatible with a broader range of models, including those designed for multi-person scenarios, while avoiding the need to adjust models that expect simpler inputs. We generate 4–10 second clips (common in existing HAR datasets [45, 50, 52]) to reduce computational demands and isolate key variables. This project is a proof-of-concept, so we prioritize simplicity over modeling complex multi-actor scenes. The types of actions are limited to all the labels provided by BEDLAM, and in our case for Kinetics [45] that works out well because they have similar labels, and, for other benchmark datasets like UCF101 [50] and HMDB51 [52] there are also matching labels with BEDLAM. By constraining our dataset to short, single-actor actions captured from a fixed camera, we strike a balance between experimental control and broad applicability, allowing us to focus on the core objective of evaluating ethical biases in action recognition systems without introducing unnecessary complexity, while generating realistic data.

Visual characteristics constraints. Each synthetic video includes seven versions of the same action, differing only in the actor's skin texture, based on Meshcapade's seven-category classification [42]. This allows us to isolate the effect of skin tone on model predictions through controlled interventions. We use a fixed camera to ensure that the only motion comes from the actor. Initial tests showed that background and camera angle affect recognition, likely due to visual similarities within action class. We ran ablations to select the best-performing viewpoint and background for each action. This minimalist setup not only simplifies interpretation but also lays the groundwork for future studies involving more variation, such as clothing, body type, or dynamic cameras.

3.2 Our synthetic dataset: Ctrl-A-Bias

To systematically evaluate model robustness to appearance-related biases in action recognition, we constructed the Ctrl-A-Bias synthetic dataset using the BEDLAM [14] framework to generate motions in the SMPL-X [39, 43] body model format, and rendered the videos in Unreal Engine. We used Python scripts to partially automate the pipeline and generate rendered video sequences, and the code is publicly available[3]. Each video in the dataset corresponds to a single row in the accompanying CSV file, which includes metadata for reproducibility. Ctrl-A-Bias contains 8,400 short videos, created by independently varying five key dimensions:

- *Skin color*: 7 texture categories from Meshcapade [42].
- *Action category*: 20 human actions matched between labels of BEDLAM and Kinetics-400 [45], see subsection 3.3.
- *Motion variant*: 10 motion clips per action, from the ones available from BEDLAM [14]. Varied clothing texture when too few motions were available per action type.
- *Camera viewpoints*: 2 fixed camera angles (near and far).
- *Background*: 3 HDRI images from Poly Haven[4].

[3]Code: https://github.com/ana-baltaretu/bias-action-recognition
[4]Poly Haven https://polyhaven.com/hdris

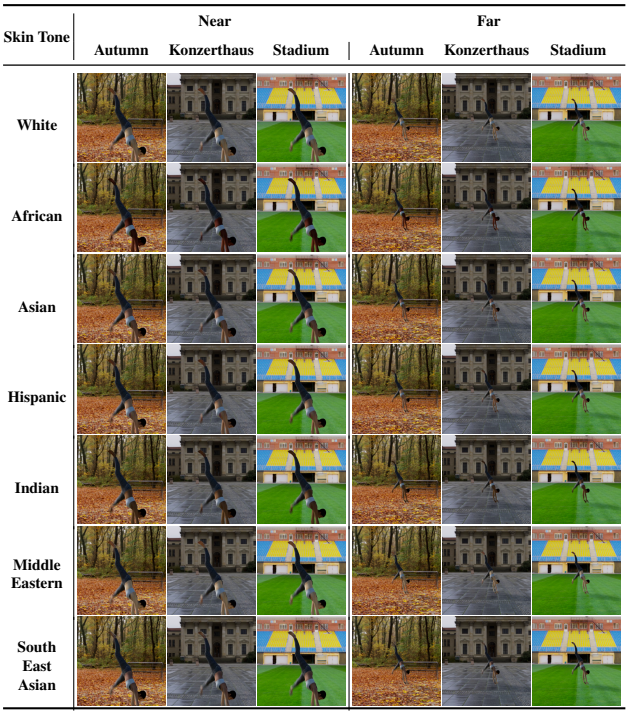


Figure 2: One motion of the cartwheel action, we see the same frame over all the different settings. This shows there is only the controlled difference we introduce in the synthetic data across individual attribute changes.

This design ensures a dataset where each combination appears exactly once. The videos labeled "initial" are used to determine representative camera angles and background variations for the five HAR models described in subsection 2.4. Based on this analysis, we select the best-performing viewpoint and background, and use the corresponding filtered subset of the dataset for bias evaluation. Our results show that model accuracy alone does not capture the full picture.

3.3 Dataset generation pipeline

Action-recognition models inherit the biases of the data on which they are trained. Public video datasets such as Kinetics-400 offer remarkable scale, but they supply little control over sensitive visual attributes like skin tone. To probe and understand these biases, we propose a synthetic-dataset generation pipeline.

1. We match labels between Kinetics [45] and BEDLAM [14] semantically, using SBERT [56].
2. Using the list of most semantically matching labels, we randomly select multiple motions per action label, along with a random body type, which has associated clothing and clothing textures.
3. We select one skin texture as part of the "initial" dataset, and we ran ablation studies to measure the influence of background and viewpoint on the model's accuracy for that action label.
4. From these, we select the "best" background and viewpoint per action label, and we apply the remaining 6 skin textures to the animation, resulting in a total 7 videos where the actor performs the exact same motion, as seen in Figure 2.
5. We compare the model accuracy across videos with the same motion where the skin color was altered. For an unbiased model, we expect the change in skin color to not affect the output labels.

4 Results

Since we do not fine-tune the models, it is essential that our synthetic data closely resembles the distributions found in the datasets the models were originally trained on. To generate realistic and representative synthetic data, we made several design choices, focusing on factors that have been shown to significantly affect model performance [57, 58, 59, 60]: camera viewpoint and background environment, see subsection 3.1. We conducted ablation studies to assess how changes in camera position and background influence accuracy. Crucially, if a model fails to understand the scene, any fairness analysis would be irrelevant, because there would be too much random variation. Therefore we only included models that met a minimum performance threshold. Lastly, we examined how sensitive each selected model is to changes in skin color and tested whether the differences in predictions were statistically significant across skin tones.

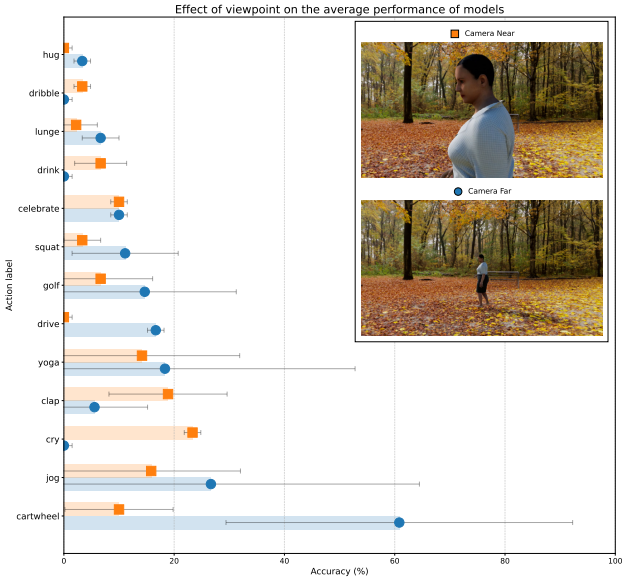


Figure 3: Impact of Viewpoint on action recognition accuracy. Mean accuracy and standard deviation over models for each action class is shown from "Near" and "Far" camera positions. Complex, full-body actions (cartwheel, jog, yoga) lose accuracy at the Near viewpoint, while simple, localized actions (drink, cry) show the opposite trend. This demonstrates that camera placement can significantly impact model evaluation especially for complex actions and highlights the necessity of using representative viewpoints when generating synthetic data for testing.

4.1 Minimizing confounding factors

To keep the scale of our synthetic benchmark manageable, we fixed most scene parameters. Every clip shows a single actor who begins the motion at screen center under uniform Unreal-Engine lighting. For factors without an obvious default (viewpoint and background) we first ran ablation studies, then picked the setting that yielded the most stable model performance.

Camera position variation. To isolate viewpoint sensitivity we rendered every motion from two viewpoints: Near (around 1m from the actor, waist-up framing) and Far (around 6m away, full-body framing). These settings mirror tight and wide shots, allowing direct comparison to real datasets. As can be seen in Figure 3, changing the camera viewpoint can shift top-1 accuracy significantly, and the direction of the effect is action-specific. Full-body activities suffer when viewed up close, because limbs leave the frame and temporal cues are lost, whereas compact upper-body actions benefit from a closer viewpoint.

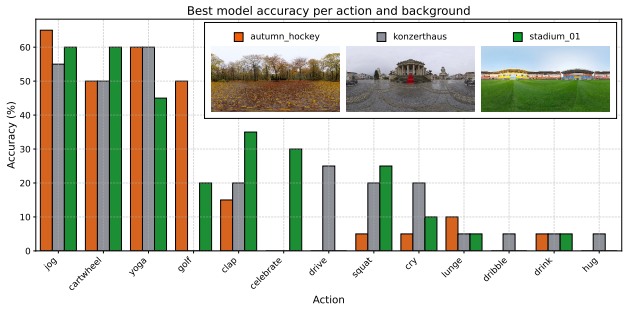


Figure 4: Impact of Background on the accuracy of models to recognize certain actions. The height of the bars represents the highest prediction accuracy of a background for any of the 5 evaluated models for each action label. We can see from this graph that some actions like jog and cartwheel have similar accuracies no matter the background, while accuracies of other actions like golf or celebrate are significantly impacted if the background differs from the training data.

Background variation. To evaluate the influence of background on model predictions, we rendered each action across three outdoor scenes: an autumn park, a grey urban plaza, and a bright stadium, each with a distinct color palette. As backgrounds occupy most of the frame, we expected them to act as strong visual cues. The results in Figure 4 confirm this: full-body motions like jog and cartwheel remain accurate across settings, while context-driven actions like golf or celebrate vary significantly by background. Some scenes, like the autumn park, consistently biased models toward a small set of actions (e.g., golf), regardless of the actual motion. To control for this bias, we fixed each action to the background with the most reliable accuracy before varying other factors.

These findings confirm that non-motion visual cues such as viewpoint and background environment affect model performance and must be controlled when probing for biases.

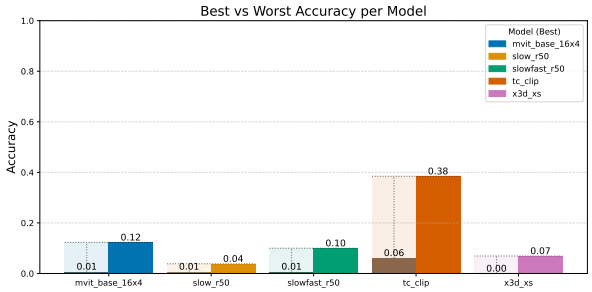


Figure 5: Model performance on the baseline synthetic dataset. For each model, the left bar shows the average accuracy under suboptimal viewpoint and background settings, while the right bar shows the accuracy with the best-performing configuration. The results highlight that selecting more suitable settings per action label improves accuracy for all models, making our baseline more comparable to real datasets.

4.2 How do models generalize to synthetic actions?

Before evaluating fairness, we first validated whether each model could reliably interpret our synthetic setup. This ensures that any observed biases are not simply due to models not understanding the actions in the first place. For this, we selected one viewpoint and one background per action label, the ones that yielded the best average accuracy across models in the ablation studies. Figure 5 shows the performance of each model under the "best" and "worst" case attribute combinations. All models improve with more favorable conditions, confirming that our synthetic design allows models to generalize to some extent. However, certain models such as `Slow_R50` and `X3D_XS` still perform poorly, even under improved settings. Their near-random accuracy suggests they lack a meaningful understanding of the actions, making any bias evaluation unreliable. To ensure a fair comparison in later experiments, we only continue evaluation on models `MViT`, `SlowFast`, and `TC-Clip`, which demonstrated better accuracy than the others and showed consistent predictions on the synthetic data.

4.3 How does skin tone affect predictions?

To assess the extent to which models rely on visual appearance (specifically skin color), rather than relying motion, we compute the prediction divergence rate between pairs of skin colors (s_1, s_2). For a given pair of skin textures, we define the divergence as the number of video instances for which the predicted action label differs when the same motion is performed by actors of skin color s_1 vs s_2, and we calculate divergence rate formally as:

$$\text{Divergence Rate}(s_1, s_2) = \frac{\sum_{i=1}^{N} y_i^{s_1} \neq y_i^{s_2}}{N} \quad (1)$$

Where:

- y is the predicted label for a video
- $(s_1,\ s_2)$ are paired videos showing the same motion i with different skin colors
- N is the total number of motions

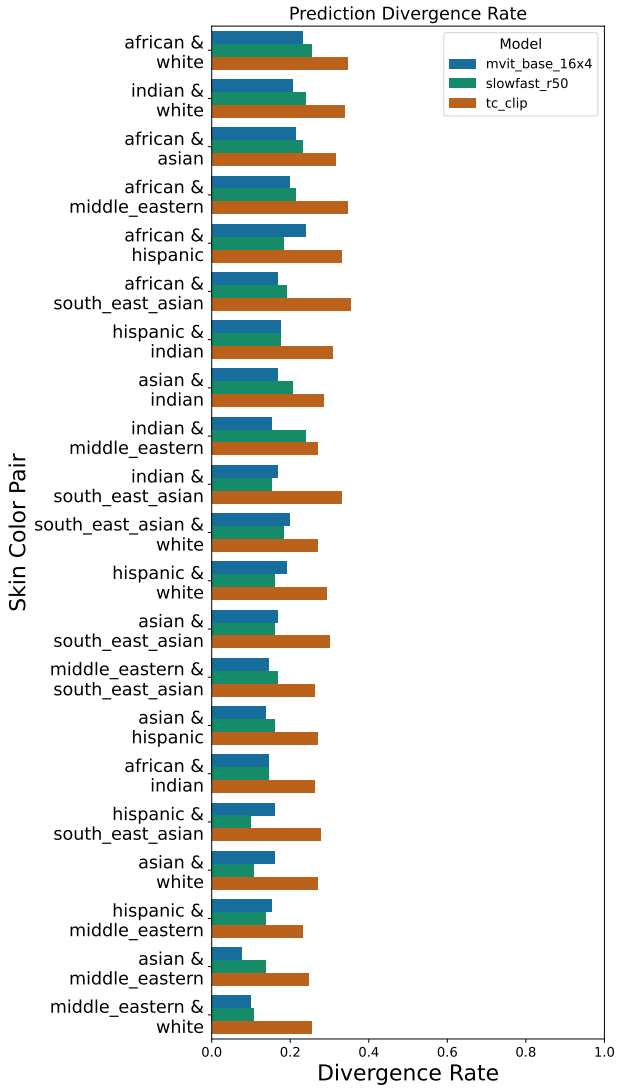


Figure 6: Proportion of action label predictions that differ when an action is performed in the same setting by actors with different skin tones. A lower divergence rate suggests less reliance on visual cues like skin color, an ideal trait when the task depends on motion rather than appearance. Notably, TC-CLIP consistently exhibits higher divergence rates across all pairs, indicating greater sensitivity to skin tone despite achieving higher average accuracy on the baseline synthetic dataset. This suggests that improved accuracy does not necessarily imply improved fairness.

This metric assesses how frequently the model changes its prediction solely due to a change in skin tone, as motion (not appearance) should be the primary cue for classification. Figure 6 shows the divergence rates across various skin color pairs for three models. Ideally, a model that is invariant to appearance would exhibit a divergence rate close to zero for all pairs. However, we observe that TC-CLIP, despite achieving the highest overall accuracy on the synthetic dataset (Figure 5), consistently demonstrates the highest divergence rates across all skin color pairs. This suggests that its predictions are more sensitive to variations in skin tone than the other two models. These results highlight that accuracy may come at the cost of fairness, especially if it is partly driven by reliance on appearance-based cues.

While all models show some variability across skin tone pairs, a truly biased model would demonstrate significantly higher divergence between a particular skin color combinations compared to the other pairs, because that means it consistently relies on the skin tone to predict actions. These outliers would point to model confusion being driven disproportionately by specific appearance factors, which we investigate further in the next section.

4.4 Do any skin tone pairs cause significantly more prediction changes?

To investigate whether models are disproportionately affected by specific skin color modifications, we statistically compare divergence rates between all skin tone pairs, shown in Figure 7. As noted earlier, a biased model would show significantly higher prediction changes for certain skin color pair, pointing to a reliance on visual cues over motion. In the raw p-values (top row), we observe several pairs that reach significance thresholds, suggesting bias in models. However, once Bonferroni correction [61] is applied to control for multiple comparisons (bottom row), nearly all significant results disappear. This outcome suggests that, while some skin tone changes may lead to more prediction changes than others, the evidence is not strong enough to confirm consistent bias toward particular skin colors. One possible explanation is that the models are generally appearance-sensitive but not selectively biased. Another explanation is that the dataset shows limited realism, not enough to expose more systematic patterns of bias.

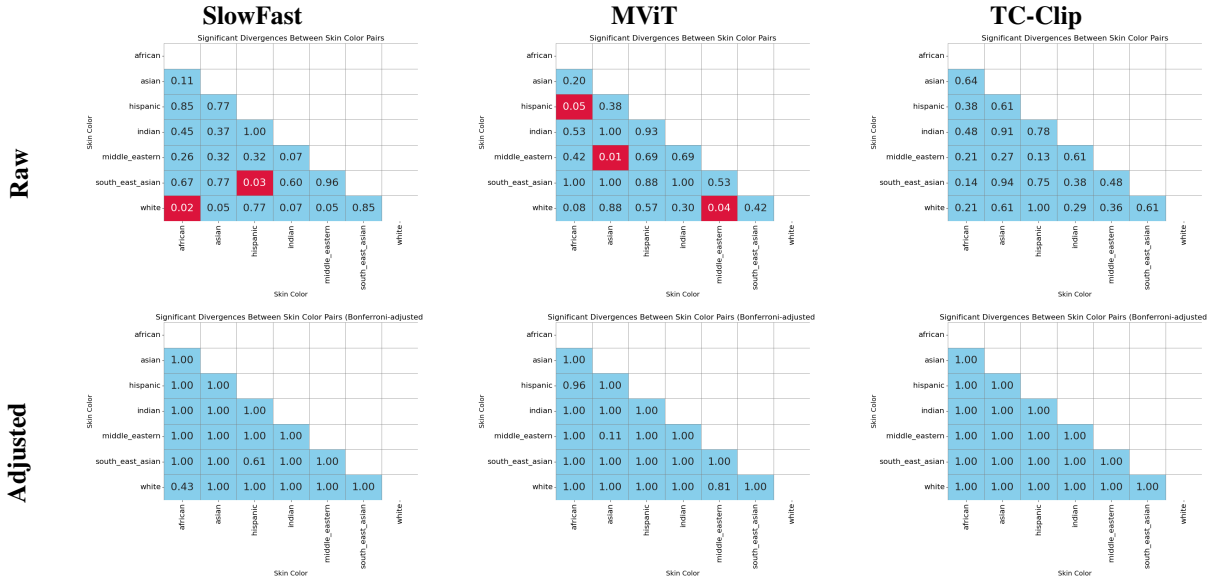


Figure 7: Statistical significance of prediction divergence between skin color pairs. Top row: raw p-values for each model and skin color pairs. Red cells indicate that prediction changes are significant ($p < 0.05$). Bottom row: Bonferroni-adjusted p-values correcting for multiple comparisons. After correction, most significant effects disappear. This may suggest that the models are not systematically biased toward specific skin color pairs, though it is also possible that such bias was not detected due to limitations in data realism.

5 Discussion

We reflect on key ethical and practical limitations of our approach and outline future directions, including issues of identity representation, potential misuse, and dataset scope.

5.1 Ethical considerations

This work involves sensitive aspects of identity and fairness in action recognition. We reflect on the ethics of using synthetic human models for auditing, focusing on harmful labels, potential misuse, and the intent of our analysis.

Demographic labels. Although our dataset includes labels such as "female" or "white" we acknowledge that these are simplifications that may reinforce binary or essentialist understandings of gender, race, or identity, which we do not support. Ideally, demographic variation would be modeled along continuous, multidimensional axes. Unfortunately, the SMPL [39, 43] framework offers only a limited set of discrete body shapes: male, female, and a neutral based on biological sex, and skin textures with categorical labels. We use these constraints to approximate demographic variation and show potential biases in model behavior, but we want to the reader to be aware of their interpretive limitations.

Risk of misuse and clarification of intended purpose. We recognize that tools for bias auditing can be misused in harmful ways in downstream applications, such as ranking individuals by demographics in surveillance or advertising. We do not support such uses. Our dataset is not for profiling, rather our goal is to enable transparent testing of models to uncover disparities in performance across demographic attributes.

While some people may view measuring bias without proposing mitigation strategies as ethically shallow, we see this work as a foundational step. Our goal is to surface and quantify disparities in model behavior, laying the groundwork for future research to develop and explore corrective measures.

5.2 Limitations and Future work

Our project enables structured auditing of action recognition models with synthetic data, but is limited by the scale of ablations and motion diversity. We outline these constraints and suggest future improvements.

More ablations. We focused on background and viewpoint variation, but other factors like lighting, action speed, actor position, and number of people may also affect model performance (subsection 2.1). These dimensions could be explored through similar ablation studies. However, we were limited to two ablations due to the significant time required to generate the data. Specifically, we produced six batches of 1400 videos each (one batch per background-viewpoint combination), with each batch taking 5–6 hours to generate. In addition to the generation time, inference on all five models added another 6 hours to the total generation time. Given these constraints, we were unable to scale up to more dimensions. Future work could explore additional ablations across the other relevant factors mentioned above, and we recommend testing each property independently, to not exponentially expand the dataset.

Limited motions. Our current setup relies on baked animations from BEDLAM [14], which are sourced from the AMASS dataset [44]. While this provides a broad set of motions, it remains limited to predefined actions and may not generalize well to specific application domains. One promising direction for future work is to leverage Meshcapade [42], which allows generating custom motion sequences through its motion-to-text feature tailored to

specific tasks or datasets. This could enable the creation of more targeted and varied ablations, especially when aligning synthetic motions with the action categories present in downstream benchmarks. We did not pursue this option in the current project for several reasons, one of them is that it lacks clothing realism and requires integration with BEDLAM's clothing pipeline[5], which was too complex for this project. Still, it holds promise for generating diverse and detailed motions in future work.

6 Conclusions

We presented a framework for auditing Human Action Recognition (HAR) models using synthetic video data with controlled appearance variations. By isolating attributes like skin tone, we evaluated whether models rely on visual cues unrelated to motion, and some key findings include:

- Viewpoint and background strongly affect model accuracy.
- Only a subset of models generalized well to synthetic data.
- All models showed sensitivity to skin tone changes.
- No significant bias between specific skin tone pairs was found after correction.

Our results highlight that appearance can still influence predictions even in the absence of bias towards a certain group. We encourage researchers and developers to use synthetic interventions to evaluate their models, especially in light of regulations like the EU AI Act [2]. While we focused on Human Action Recognition, the presented methodology could also be applied to other video-based tasks such as video understanding or video captioning. We see this work as a first step: a foundation on which more domain-specific bias auditing tools and mitigation strategies can be built.

[5]BEDLAM clothing repository: https://github.com/PerceivingSystems/bedlam_clothing

References

[1] Lanfei Zhao et al. "A Review of State-of-the-Art Methodologies and Applications in Action Recognition". In: *Electronics* 13.23 (2024), p. 4733.

[2] LAYING DOWN and INTELLIGENCE ACT. "Proposal for a Regulation of the European Parliament and of the Council laying down harmonised rules on Artificial Intelligence (Artificial Intelligence Act) and amending certain Union legislative acts". In: (2021). https : / / eur - lex . europa . eu / legal - content/EN/TXT/?uri=CELEX:52021PC0206.

[3] Lilian Edwards. "The EU AI Act: a summary of its significance and scope". In: *Artificial Intelligence (the EU AI Act)* 1 (2021).

[4] Filip Ilic, Thomas Pock, and Richard P Wildes. "Is appearance free action recognition possible?" In: *European Conference on Computer Vision*. Springer. 2022, pp. 156–173.

[5] Tuan-Hung Vu et al. "Predicting actions from static scenes". In: *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*. Springer. 2014, pp. 421–436.

[6] Yun He et al. "Human action recognition without human". In: *Computer Vision–ECCV 2016 Workshops: Amsterdam, The Netherlands, October 8-10 and 15-16, 2016, Proceedings, Part III 14*. Springer. 2016, pp. 11–17.

[7] Jinwoo Choi et al. "Why can't i dance in the mall? learning to mitigate scene bias in action recognition". In: *Advances in Neural Information Processing Systems* 32 (2019).

[8] Jihoon Chung, Yu Wu, and Olga Russakovsky. "Enabling detailed action recognition evaluation through video dataset augmentation". In: *Advances in Neural Information Processing Systems* 35 (2022), pp. 39020–39033.

[9] European Union. *Charter of Fundamental Rights of the European Union*. Dec. 2000. URL: https://www.europarl.europa.eu/charter/pdf/text_en.pdf.

[10] Joy Buolamwini and Timnit Gebru. "Gender shades: Intersectional accuracy disparities in commercial gender classification". In: *Conference on fairness, accountability and transparency*. PMLR. 2018, pp. 77–91.

[11] Tianlu Wang et al. "Balanced datasets are not enough: Estimating and mitigating gender bias in deep image representations". In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2019, pp. 5310–5319.

[12] Hao Liang, Pietro Perona, and Guha Balakrishnan. "Benchmarking algorithmic bias in face recognition: An experimental approach using synthetic faces and human evaluation". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2023, pp. 4977–4987.

[13] Nikita Kister et al. “Are Pose Estimators Ready for the Open World? STAGE: Synthetic Data Generation Toolkit for Auditing 3D Human Pose Estimators”. In: (2024).

[14] Michael J Black et al. “Bedlam: A synthetic dataset of bodies exhibiting detailed lifelike animated motion”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 8726–8737.

[15] Leander Girrbach et al. “A Large Scale Analysis of Gender Biases in Text-to-Image Generative Models”. In: *arXiv preprint arXiv:2503.23398* (2025).

[16] Jen-tse Huang et al. “VisBias: Measuring Explicit and Implicit Social Biases in Vision Language Models”. In: *arXiv preprint arXiv:2503.07575* (2025).

[17] Shresth Grover, Vibhav Vineet, and Yogesh Rawat. “Revealing the unseen: Benchmarking video action recognition under occlusion”. In: *Advances in Neural Information Processing Systems* 36 (2023), pp. 65642–65664.

[18] Madeline Chantry Schiappa et al. “A large-scale robustness analysis of video action recognition models”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2023, pp. 14698–14708.

[19] Matias Duran et al. “Metamorphic Testing for Pose Estimation Systems”. In: *arXiv preprint arXiv:2502.09460* (2025).

[20] Xingrui Wang et al. “PulseCheck457: A Diagnostic Benchmark for 6D Spatial Reasoning of Large Multimodal Models”. In: *arXiv e-prints* (2025), arXiv–2502.

[21] Gyeongsik Moon et al. “Integralaction: Pose-driven feature integration for robust human action recognition in videos”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2021, pp. 3339–3348.

[22] Jain Liu, Naveed Akhtar, and Ajmal Mian. “Viewpoint invariant RGB-D human action recognition”. In: *2017 International Conference on Digital Image Computing: Techniques and Applications (DICTA)*. IEEE. 2017, pp. 1–8.

[23] Manoj Ramanathan, Wei-Yun Yau, and Eam Khwang Teoh. “Human action recognition with video data: research and evaluation challenges”. In: *IEEE Transactions on Human-Machine Systems* 44.5 (2014), pp. 650–663.

[24] Yogesh Singh Rawat and Shruti Vyas. “View-invariant action recognition”. In: *Computer Vision: A Reference Guide*. Springer, 2021, pp. 1341–1341.

[25] Gül Varol et al. “Synthetic humans for action recognition from unseen viewpoints”. In: *International Journal of Computer Vision* 129.7 (2021), pp. 2264–2287.

[26] Fernando Camarena et al. “An overview of the vision-based human action recognition field”. In: *Mathematical and Computational Applications* 28.2 (2023), p. 61.

[27] Andrea Zunino, Jacopo Cavazza, and Vittorio Murino. “Revisiting human action recognition: Personalization vs. generalization”. In: *Image Analysis and Processing-ICIAP 2017: 19th International Conference, Catania, Italy, September 11-15, 2017, Proceedings, Part I 19*. Springer. 2017, pp. 469–480.

[28] Anna Ferrari et al. “Personalization in human activity recognition”. In: *arXiv preprint arXiv:2009.00268* (2020).

[29] https://deepmind.google/models/veo/. 2025.

[30] Tim Brooks et al. “Video generation models as world simulators. 2024”. In: 3 (2024). https://openai.com/research/video-generation-models-as-world-simulators, p. 1.

[31] 2024. URL: https://runwayml.com/research/introducing-runway-gen-4.

[32] Adam Polyak et al. *Movie Gen: A Cast of Media Foundation Models*. 2025. arXiv: 2410.13720 [cs.CV]. URL: https://arxiv.org/abs/2410.13720.

[33] Hong-Bo Zhang et al. “A comprehensive survey of vision-based human action recognition methods”. In: *Sensors* 19.5 (2019), p. 1005.

[34] David Griffiths and Jan Boehm. “SynthCity: A large scale synthetic point cloud”. In: *arXiv preprint arXiv:1907.04758* (2019).

[35] Suncheng Xiang et al. “Taking a closer look at synthesis: Fine-grained attribute analysis for person re-identification”. In: *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2021, pp. 3765–3769.

[36] Erroll Wood et al. “Fake it till you make it: face analysis in the wild using synthetic data alone”. In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2021, pp. 3681–3691.

[37] Yana Hasson et al. “Learning joint reconstruction of hands and manipulated objects”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2019, pp. 11807–11816.

[38] Jinhyeok Jang et al. “ETRI-activity3D: A large-scale RGB-D dataset for robots to recognize daily activities of the elderly”. In: *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2020, pp. 10990–10997.

[39] Matthew Loper et al. “SMPL: A Skinned Multi-Person Linear Model”. In: *ACM Trans. Graphics (Proc. SIGGRAPH Asia)* 34.6 (Oct. 2015), 248:1–248:16.

[40] Abhinanda R Punnakkal et al. “BABEL: Bodies, action and behavior with english labels”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021, pp. 722–731.

[41] David Schneider et al. “Synthact: Towards generalizable human action recognition based on synthetic data”. In: *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2024, pp. 13038–13045.

[42] 2018. URL: https://meshcapade.com/.

[43] Georgios Pavlakos et al. “Expressive Body Capture: 3D Hands, Face, and Body from a Single Image”. In: *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2019, pp. 10975–10985.

[44] Naureen Mahmood et al. “AMASS: Archive of Motion Capture as Surface Shapes”. In: *International Conference on Computer Vision*. Oct. 2019, pp. 5442–5451.

[45] Will Kay et al. “The kinetics human action video dataset”. In: *arXiv preprint arXiv:1705.06950* (2017).

[46] Christoph Feichtenhofer et al. “Slowfast networks for video recognition”. In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2019, pp. 6202–6211.

[47] Haoqi Fan et al. “Multiscale vision transformers”. In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2021, pp. 6824–6835.

[48] Minji Kim et al. “Leveraging temporal contextualization for video action recognition”. In: *European Conference on Computer Vision*. Springer. 2024, pp. 74–91.

[49] Christoph Feichtenhofer. “X3d: Expanding architectures for efficient video recognition”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2020, pp. 203–213.

[50] K Soomro. “UCF101: A dataset of 101 human actions classes from videos in the wild”. In: *arXiv preprint arXiv:1212.0402* (2012).

[51] Chunhui Gu et al. “Ava: A video dataset of spatio-temporally localized atomic visual actions”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 6047–6056.

[52] Hildegard Kuehne et al. “HMDB: a large video database for human motion recognition”. In: *2011 International conference on computer vision*. IEEE. 2011, pp. 2556–2563.

[53] Raghav Goyal et al. “The” something something” video database for learning and evaluating visual common sense”. In: *Proceedings of the IEEE international conference on computer vision*. 2017, pp. 5842–5850.

[54] Fabian Caba Heilbron et al. “Activitynet: A large-scale video benchmark for human activity understanding”. In: *Proceedings of the ieee conference on computer vision and pattern recognition*. 2015, pp. 961–970.

[55] Dima Damen et al. “Rescaling egocentric vision: Collection, pipeline and challenges for epic-kitchens-100”. In: *International Journal of Computer Vision* (2022), pp. 1–23.

[56] Nils Reimers and Iryna Gurevych. “Sentence-bert: Sentence embeddings using siamese bert-networks”. In: *arXiv preprint arXiv:1908.10084* (2019).

[57] Yu Kong and Yun Fu. “Human action recognition and prediction: A survey”. In: *International Journal of Computer Vision* 130.5 (2022), pp. 1366–1401.

[58] Fei Wu et al. “A survey on video action recognition in sports: Datasets, methods and applications”. In: *IEEE Transactions on Multimedia* 25 (2022), pp. 7943–7966.

[59] Zehua Sun et al. “Human action recognition from various data modalities: A review”. In: *IEEE transactions on pattern analysis and machine intelligence* 45.3 (2022), pp. 3200–3225.

[60] Djamila Romaissa Beddiar et al. “Vision-based human activity recognition: a survey”. In: *Multimedia Tools and Applications* 79.41 (2020), pp. 30509–30555.

[61] Richard A Armstrong. “When to use the B onferroni correction”. In: *Ophthalmic and physiological optics* 34.5 (2014), pp. 502–508.

3

Additional information

In this chapter we go over additional information that did not fit anywhere else in the story, specifically looking into some experiments and methods that we tried and didn't work or that were too simplistic.

3.1. Additional experiment: Cubes

To investigate model sensitivity to visual appearance factors such as color, we designed a simplified action recognition task using synthetic 3D animations of cubes performing two visibly distinct motions: orbiting and bouncing[1,2]. The aim was to explore whether a model trained only on red and blue cubes would generalize poorly when presented with unseen colors, particularly green. We chose to train a ConvLSTM model from scratch using a public repository[3] due to its simple setup and ease of integration. The focus of this experiment was not on model architecture but rather on generating controlled synthetic data to probe for potential color bias.

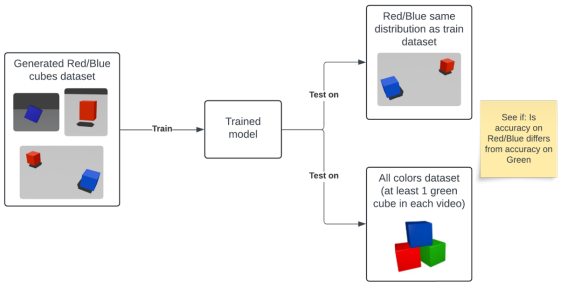


Figure 3.1: Pipeline for the cubes experiment: model is trained on red/blue-only data, then tested on balanced and green-included sets to probe color bias.

Dataset generation was fully automated in Blender and rendered on the DAIC cluster[4,5] through a single command-line. We produced 90 videos in total for training and testing, accompanied by 5 validation videos drawn from the same distribution, and an additional 5 validation videos where cube colors were systematically altered to introduce green. For the results we made sure that the validation videos were exactly the same, with the only difference being the color of some cubes being changed to green. Initial training results were surprisingly disappointing. Despite the task involving only two clearly distinct motions, the model achieved only 70% accuracy on in-distribution test and validation data, and a significant drop to 30% on the altered-color validation set (see Figure 3.2). These results suggested either insufficient training data or the presence of confounding variables, particularly, viewpoint variation. To isolate potential confounding factors, we modified the camera setup.

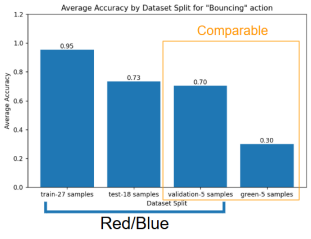


Figure 3.2: Initial results for the bouncing action. The last two charts show validation data for comparing bias effects.

[1] Orbiting and bouncing video 1: https://www.youtube.com/watch?v=F1Ofo4viVho
[2] Orbiting and bouncing video 2: https://www.youtube.com/watch?v=ZscNUexIIbk
[3] Repository for training ConvLSTM: https://github.com/eriklindernoren/Action-Recognition
[4] DAIC cluster: https://daic.tudelft.nl/
[5] Code for generating cubes dataset: https://github.com/ana-baltaretu/bias-action-recognition/tree/main/daic/parallel_cube_render

Initially, cameras were randomly placed in a hemisphere around the scene, resulting in high variability in viewpoint (Figure 3.3). To reduce this variation, we constrained camera positions to lie on a fixed plane, resulting in a dataset with a more controlled diversity in viewpoints. This intervention was associated with a significant performance boost[6], suggesting that the model's poor performance was not primarily due to color bias, but rather due to a lack of viewpoint invariance, a known limitation in other papers.

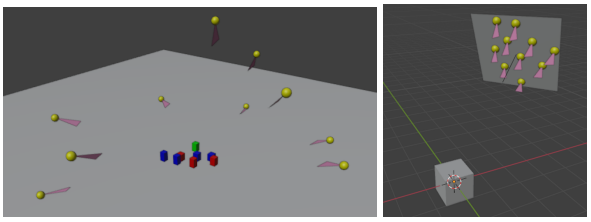


Figure 3.3: Camera setup for controlling viewpoint variation. Left: cameras are placed in a hemisphere. Right: cameras are restricted to a plane. Spheres indicate camera positions, triangles show camera orientations, cubes represent the scene.

Another key observation was that ConvLSTM outputs frame-level predictions. While this was suitable for our initial experiments, it became clear that not all action recognition models operate at frame level. Therefore, for consistency and comparability with other models, we decided to evaluate future experiments using a single action label per video.

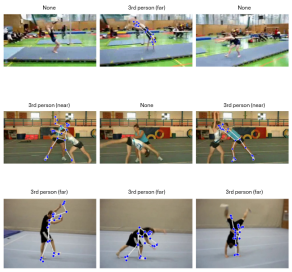
3.2. Attempted methods

To make the framework more flexible and generated synthetic data more realistic, we attempted to make some pre-generation steps that would extract information from the dataset that the models are trained on, such that we can make our dataset more similar. The following parts are just rough set-ups that did not entirely work and are not implemented in the final pipeline. Therefore we attempted to extract:

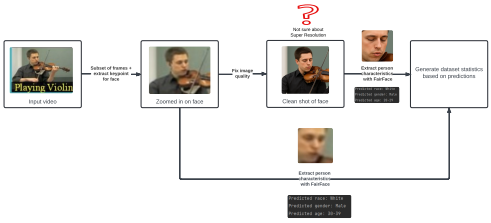
- **Viewpoint information per action label:** BEDLAM allows flexibility in camera placement, along with some preset viewpoints. However, instead of using presets, we aimed to match the viewpoints to those commonly found in the datasets the models were trained on. This aims to reduce recognition errors that occur when our test viewpoints differ too much from training data. Ideally, models would generalize well across viewpoints, making this step unnecessary, but given the current limitations of action recognition, such heuristics remain a practical necessity. To approximate this viewpoint, we used a skeleton detection model (OpenPose [39]) for estimating actor distance from the camera and visible body parts, see Figure 3.4a, and through some hard-coded heuristics by we classify viewpoint types in real videos. If a specific action like "cartwheel" most often appeared in a full-body third-person view, we selected that as the default for generating synthetic versions of that action. Unfortunately the heuristics lead to a lot of variations per action, and the workload of this part would have been a research project on its own, so we did not continue on this idea.
- **Skin color distribution over the dataset:** To better understand potential biases in action recognition models, we aimed to extract the distribution of skin colors in their training datasets. We used the FairFace model [13] for race prediction, following the pipeline illustrated in Figure 3.4b. Since datasets like Kinetics [23] and HMDB-51 [26] are relatively old (in Computer Vision standards), and the videos feature people who are not necessarily close to the camera, the quality of zoomed in faces is not great. FairFace, however, requires a close-up, high-quality facial image for accurate predictions. To address this, we selected a subset of frames from each video, identified face keypoints, and zoomed in on the face. To improve the image quality, we applied a super-resolution model before feeding it into FairFace. We also tested the pipeline without super-resolution, and, while it worked for some videos, actions involving distant actors resulted in blurry zoomed-in faces, which can degrade prediction quality. Overall, the approach was effective in extracting race information, however, we encountered a significant limitation: Kinetics is no longer fully accessible, as many YouTube videos from the original dataset have since been removed. Because we wanted our pipeline to be usable even if you do not have access to the training videos, we ultimately decided to drop this extraction step.

[6] Accuracy after fixing viewpoint: https://gitlab.ewi.tudelft.nl/in5000/janvangemert/ana-baltaretu/bias-action-recognition/-/blob/main/data/RGB_cubes/90_scenes/results_test.md

- **Clothing texture information:** The goal of this part was to inform our choice of clothing textures for actors in a baseline dataset. Initially, on top of skin textures, we planned to investigate whether clothing influences the predicted actions. This is important in practical contexts since we wouldn't want someone to be misclassified as performing an action like "robbing" simply because they are wearing a hoodie. Additionally, clothing is often associated with gender expression (e.g., more "girly" or "boyish" clothes), which raises concerns related to non-discrimination on the basis of gender or sexual orientation, as addressed by the EU Charter of Fundamental Rights [34]. Some promising prior works on 3D clothing extraction are SCARF [40] and REC-MV [41], which reconstruct 3D clothing. These methods could have been ideal, but we were unsure how easily these methods could be integrated with motions from BEDLAM. A more relevant direction for us was be [42] which extracts clothing items from videos. Inspired by this, we prototyped a pipeline using using MediaPipe's body part segmentation along with a skeleton extraction model. The idea was to segment visible clothing items by associating them with specific body parts and overlapping joints, and then match these to BEDLAM's available clothing textures. Similarly to our other extraction approaches, this path turned out to be relatively complex and, given that the idea was still underdeveloped, we decided not to pursue it further.



(a) Examples of extracted skeletons with OpenPose [39] on the HMDB-51 dataset [26], used for estimating viewpoint.



(b) Pipeline for extracting skin color distribution from the initial dataset using the FairFace [13] model.

(c) Simplified clothing extraction method with Mediapipe[7] to be able to match characters to BEDLAM [11] clothing textures.

Figure 3.4: Methods that we attempted to use to extract information from the real training datasets to aid in the generation of synthetic datasets. None of these methods were integrated in the final project.

3.3. How did we end up with this direction?

When deciding on a direction (about half way through the project timeline), we had 3 possible options:

1. Continue using the hyper-simplified cube dataset, incrementally adding more realistic attributes. This would involve fully training models from scratch and intentionally injecting biases during training to observe how they affect performance.
2. Shift to a Minecraft-based approach, leveraging its built-in physics engine and mod support to simulate realistic actions. We could fine-tune foundation models on Minecraft data and evaluate their performance on Minecraft-specific motions. However, this posed the risk of introducing unintended biases during fine-tuning.
3. Generate synthetic videos of realistic human models without any fine-tuning, allowing us to evaluate existing models in a controlled yet unbiased way.

At first, we were hesitant to pursue the third option, since generating realistic synthetic data seemed technically complex and time-consuming. However, discovering BEDLAM [11] significantly simplified this workflow. Another concern with this approach was that models might not generalize to synthetic data at all, resulting in poor or even random performance, resulting in models performing poorly or almost random (in our case that would mean an accuracy of around $\frac{1}{400}$, since the dataset we chose was Kinetics-400 [23], with 400 action labels). Despite these risks, we ultimately chose the third path: generating synthetic data and avoiding any model fine-tuning. This decision allows for the most realistic evaluation of off-the-shelf action recognition models.

[7] Mediapipe body part segmentation: https://storage.googleapis.com/mediapipe-assets/Model%20Card%20Multiclass%20Segmentation.pdf

4

Conclusion

This thesis explored a difficult question: *Are we SMPLy biased?* In short, yes, we are. Bias is inherent to the human nature, and any model trained by humans unfortunately reflects that. The goal is then, not to eliminate bias entirely, but to understand, explain, and eventually mitigate its effects, especially in high-risk AI applications such as Human Action Recognition (HAR). In this work, we presented a method for auditing HAR models using synthetic video data generated through the BEDLAM [11] framework. By controlling for visual attributes like skin tone, background, and camera viewpoint, I examined whether models rely on appearance cues unrelated to motion. The findings were nuanced: while no systematic bias was found between specific skin tone pairs after statistical correction, all models showed sensitivity to appearance-based changes, suggesting a broader problem of reliance on visual context. These findings are important to real-world deployments of HAR models, whether in surveillance, healthcare, or human-computer interaction, they fall under the EU AI Act's [2] category of High-Risk AI Systems. Current evaluation practices prioritize model accuracy on benchmarks, without looking into how the model gets to those predictions. As a result, we risk deploying models we don't fully understand and can't reliably interpret.

This work does not claim to offer a complete solution, but instead provides a systematic approach to identifying potential issues in Human Action Recognition models, serving as a foundation for future research. We highlight that if a model performs so inconsistently when small and controlled changes are made to its inputs, it is likely that it would behave unpredictably in the real-world. This work shows that we need more rigorous auditing methods for checking models before they are deployed. It is important to note that not all biases are equal. While we must actively avoid biases tied to protected attributes like skin color, gender, and disability, other biases like associating the background to the performed action might be acceptable or even necessary for accurate recognition. As a suggested approach we would define which biases are ethically and legally unacceptable, and then structure datasets accordingly. For example, ensuring equal representation of demographic attributes, while allowing contextual cues like background to remain unbalanced when appropriate.

Ultimately, HAR models reflect the world we show them, if that world is skewed, their decisions will also be biased. We cannot remove all bias, but we can improve transparency, explainability, and accountability. That starts by asking hard questions, building more controlled datasets, and not letting the pursuit of results on leaderboard benchmarks distract us from the ethical obligations of our work.

References

[1] LAYING DOWN and INTELLIGENCE ACT. “Proposal for a Regulation of the European Parliament and of the Council laying down harmonised rules on Artificial Intelligence (Artificial Intelligence Act) and amending certain Union legislative acts”. In: (2021). https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX:52021PC0206.

[2] Lilian Edwards. “The EU AI Act: a summary of its significance and scope”. In: *Artificial Intelligence (the EU AI Act)* 1 (2021).

[3] Frank Rosenblatt. “The perceptron: a probabilistic model for information storage and organization in the brain.” In: *Psychological review* 65.6 (1958), p. 386.

[4] Christoph Feichtenhofer et al. “Slowfast networks for video recognition”. In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2019, pp. 6202–6211.

[5] Christoph Feichtenhofer. “X3d: Expanding architectures for efficient video recognition”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2020, pp. 203–213.

[6] Ashish Vaswani et al. “Attention is all you need”. In: *Advances in neural information processing systems* 30 (2017).

[7] Alexey Dosovitskiy et al. “An image is worth 16x16 words: Transformers for image recognition at scale”. In: *arXiv preprint arXiv:2010.11929* (2020).

[8] Minji Kim et al. “Leveraging temporal contextualization for video action recognition”. In: *European Conference on Computer Vision*. Springer. 2024, pp. 74–91.

[9] Haoqi Fan et al. “Multiscale vision transformers”. In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2021, pp. 6824–6835.

[10] Matthew Loper et al. “SMPL: A Skinned Multi-Person Linear Model”. In: *ACM Trans. Graphics (Proc. SIGGRAPH Asia)* 34.6 (Oct. 2015), 248:1–248:16.

[11] Michael J Black et al. “Bedlam: A synthetic dataset of bodies exhibiting detailed lifelike animated motion”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 8726–8737.

[12] 2018. URL: https://meshcapade.com/.

[13] Kimmo Karkkainen and Jungseock Joo. “Fairface: Face attribute dataset for balanced race, gender, and age for bias measurement and mitigation”. In: *Proceedings of the IEEE/CVF winter conference on applications of computer vision*. 2021, pp. 1548–1558.

[14] Georgios Pavlakos et al. “Expressive Body Capture: 3D Hands, Face, and Body from a Single Image”. In: *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2019, pp. 10975–10985.

[15] Naureen Mahmood et al. “AMASS: Archive of motion capture as surface shapes”. In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2019, pp. 5442–5451.

[16] Tamara Munzner. *Visualization analysis and design*. CRC press, 2014.

[17] Stuart K Card, Jock Mackinlay, and Ben Shneiderman. *Readings in information visualization: using vision to think*. Morgan Kaufmann, 1999.

[18] Li Fei-Fei, Stanford Fergus, and Antonio Torralba. *Recognizing and Learning Object Categories*. https://www.cs.bilkent.edu.tr/~duygulu/Courses/CS554/Notes/ObjectRecognition-Intro.pdf.

[19] Abhinanda R Punnakkal et al. “BABEL: Bodies, action and behavior with english labels”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021, pp. 722–731.

[20] Filip Ilic, Thomas Pock, and Richard P Wildes. “Is appearance free action recognition possible?” In: *European Conference on Computer Vision*. Springer. 2022, pp. 156–173.

[21] Tuan-Hung Vu et al. “Predicting actions from static scenes”. In: *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*. Springer. 2014, pp. 421–436.

[22] Jinwoo Choi et al. “Why can’t i dance in the mall? learning to mitigate scene bias in action recognition”. In: *Advances in Neural Information Processing Systems* 32 (2019).
[23] Will Kay et al. “The kinetics human action video dataset”. In: *arXiv preprint arXiv:1705.06950* (2017).
[24] Minji Kim et al. “Leveraging temporal contextualization for video action recognition”. In: *European Conference on Computer Vision*. Springer. 2024, pp. 74–91.
[25] K Soomro. “UCF101: A dataset of 101 human actions classes from videos in the wild”. In: *arXiv preprint arXiv:1212.0402* (2012).
[26] Hildegard Kuehne et al. “HMDB: a large video database for human motion recognition”. In: *2011 International conference on computer vision*. IEEE. 2011, pp. 2556–2563.
[27] Chunhui Gu et al. “Ava: A video dataset of spatio-temporally localized atomic visual actions”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 6047–6056.
[28] Raghav Goyal et al. “The” something something” video database for learning and evaluating visual common sense”. In: *Proceedings of the IEEE international conference on computer vision*. 2017, pp. 5842–5850.
[29] Dima Damen et al. “Rescaling egocentric vision: Collection, pipeline and challenges for epic-kitchens-100”. In: *International Journal of Computer Vision* (2022), pp. 1–23.
[30] Gunnar A Sigurdsson et al. “Hollywood in homes: Crowdsourcing data collection for activity understanding”. In: *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14*. Springer. 2016, pp. 510–526.
[31] Yingwei Li, Yi Li, and Nuno Vasconcelos. “Resound: Towards action recognition without representation bias”. In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, pp. 513–528.
[32] Jia Deng et al. “Imagenet: A large-scale hierarchical image database”. In: *2009 IEEE conference on computer vision and pattern recognition*. Ieee. 2009, pp. 248–255.
[33] Alec Radford et al. “Learning transferable visual models from natural language supervision”. In: *International conference on machine learning*. PmLR. 2021, pp. 8748–8763.
[34] European Union. *Charter of Fundamental Rights of the European Union*. Dec. 2000. URL: https://www.europarl.europa.eu/charter/pdf/text_en.pdf.
[35] 2016. URL: https://eur-lex.europa.eu/eli/reg/2016/679/oj/eng.
[36] Hans-W Micklitz and Giovanni Sartor. “Compliance and enforcement in the AIA through AI”. In: *Yearbook of European Law* (2025), yeae014.
[37] Martin Ebers. “Truly risk-based regulation of artificial intelligence how to implement the EU’s AI Act”. In: *European Journal of Risk Regulation* (2024), pp. 1–20.
[38] Jeffrey Dastin. *Insight - Amazon scraps secret AI recruiting tool that showed bias against women*. Oct. 2018. URL: https://www.reuters.com/article/world/insight-amazon-scraps-secret-ai-recruiting-tool-that-showed-bias-against-women-idUSKCN1MK0AG/?utm_source=chatgpt.com.
[39] Zhe Cao et al. “Realtime multi-person 2d pose estimation using part affinity fields”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 7291–7299.
[40] Yao Feng et al. “Capturing and animation of body and clothing from monocular video”. In: *SIGGRAPH Asia 2022 Conference Papers*. 2022, pp. 1–9.
[41] Lingteng Qiu et al. “Rec-mv: Reconstructing 3d dynamic cloth from monocular videos”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 4637–4646.
[42] Noa Garcia and George Vogiatzis. “Dress like a star: Retrieving fashion products from videos”. In: *Proceedings of the IEEE International Conference on Computer Vision Workshops*. 2017, pp. 2293–2299.
[43] Chun-Fu Richard Chen et al. “Deep analysis of cnn-based spatio-temporal representations for action recognition”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2021, pp. 6165–6175.

A

Tools and technologies

In this section we go over more details on the tools and technologies we used in the project, and the training dataset.

Blender. Animations from AMASS [15] were baked into Alembic (.abc) files using Blender, following the BEDLAM guide[1]. Unlike .npz files which are normally used to share SMPL [10, 14] animations, alembic files store fixed vertex positions and are larger, but render faster, which is especially useful since we re-render the same motions multiple times.

Unreal Engine, used for rendering to stay consistent with BEDLAM's setup. Despite having a headless mode, Unreal is tedious to configure, making the BEDLAM set-up manual and time-consuming. This also makes the dataset harder to generate than alternatives like using Blender scripts.

Hardware specifications. We used Python 3.10 for Blender scripting (converting SMPL to Alembic), scene and animation sequence file generation (`be_seq.csv`). Rendering was done in Unreal Engine 5.0.3 on Windows 11. The system ran on an Intel i9-14900KF CPU (for fast simulation and asset processing), an NVIDIA RTX 4080 GPU, and 128GB RAM (required by Unreal for character-heavy scenes). At least 3TB of storage is needed due to large Alembic (`.abc`) files and rendered frame outputs.

Training Data. Kinetics-400 [23] serves as a standard benchmark for evaluating HAR models [43]. It is a large-scale video dataset containing 400 action classes and over 306,245 clips, each lasting around 10 seconds, sourced from a diverse set of unedited YouTube videos. This diversity makes Kinetics-400 valuable for training models in realistic conditions, as the data reflects natural human behavior. Additionally, many of its action labels overlap semantically with those found in motion capture datasets like AMASS [15], supporting its relevance to this work. The availability of pretrained model weights was a key factor in our study's choice to utilize Kinetics-400 since it facilitates the execution of our experiments. While we also considered HMDB51 [26] and UCF101 [25], both commonly used in HAR research, these datasets are smaller in scale and are more commonly used for evaluation rather than training. This means that less pretrained model weights were available. Kinetics-400 provided the simplest and most effective way forward because our main goal was to create and assess synthetic datasets rather than modify model architectures or train models. Because of its scale, diversity, and benchmarking role, Kinetics-400 provided a solid foundation for evaluating model behavior on our synthetic test sets.

[1] Bedlam guide: https://bedlam.is.tue.mpg.de/index.html

B

Acknowledgment of AI assistance

Throughout this project, I made use of AI tools, specifically large language models (LLMs) to speed up tasks and have more time to focus on the larger picture. My general approach was to provide them with as much context as possible, prompt them repeatedly with variations of the same request, and then manually review and combine the outputs. *Cherry-picking and critically evaluating the outputs are mandatory skills to have when using AI.* I used the following tools:

- ChatGPT: https://chatgpt.com/
- Quillbot: https://quillbot.com/paraphrasing-tool

Understanding background concepts. When starting a new project, I often find the terminology and mathematical notation overwhelming. It can be hard to follow if formulas are complex or if terminology is used inconsistently within or across papers. ChatGPT was incredibly helpful here: it explains difficult concepts using comparisons, simpler language, or real-life analogies, which aligns with how I naturally like to learn and explain things. It was also helpful for summarizing papers to quickly decide if they were worth a deeper look.

Idea exploration and brainstorming. I used the voice feature to talk through ideas, which helped me think more clearly. I'm more inclined to ask questions than to seek direct answers, and this interactive conversation clarified my thoughts.

Deep research with ChatGPT. A newer feature I used (starting February 3, 2025) let me conduct focused research queries. I would specify a topic, and it would ask me questions to help clarify my intent and retrieve relevant papers, sort of like a research assistant.

"Vibe" coding. I've reached a point where I can read code in any language, but switching between languages can still be tedious. I used LLMs to generate small, clearly-defined functions after I had already mapped out the code structure and purpose of each method in my head. Writing the logic in natural language feels more intuitive and it allowed me to describe the idea while offloading the syntax. This process fits with my coding style, where I typically write and test one function at a time before moving on to the next. While this workflow is effective, it also requires thorough code review. LLMs are prone to generating bugs, and re-prompting rarely resolves them fully. In practice, I chose to debug and fix issues myself rather than relying on the model to correct its own output.

Writing and Rephrasing I love explaining my thoughts, and visualizing them, but my writing tends to be informal. Quillbot helped preserve the meaning of a sentence while changing its structure, which was great for avoiding repetition, while ChatGPT, was better for condensing explanations. Together, they helped me express my thoughts more clearly and concisely without compromising meaning.

Looking back three years ago, I remember how much time I spent just trying to understand the research question for my Research Project (before ChatGPT and other LLMs became widely available). Having access to this kind of tools allowed me to shift focus toward the aspects I found most meaningful.