1. **Работа с данными**
   1. **Формат данных**

Датасет содержит набор папок, каждая папка – отдельный класс. В каждом классе содержится по одному файлу .pkl для каждого видео. Пусть N — количество кадров в одном видео. Тогда файл .pkl содержит карту со следующими ключами:

* time: Массив размером N, где каждый элемент — это время в секундах с начала 10-секундного клипа (а не с начала всего видео)
* 2d\_keypoints: Массив размером Nx19x2, где каждый срез представляет собой 19 сочленений, спроецированных из модели SMPL с помощью пользовательского регрессора ключевых точек, описанного ниже, в координатах (x,y). Эти координаты нормализованы относительно рамки изображения: таким образом, (0, 0) и (1, 1) — это верхний левый и нижний правый углы соответственно.
  1. **Параметры данных**

В основном видео в наборе имеют от 0 до 27 fps. Большее количество сосредоточено в пределах 26 fps. В дальнейшем эта модель будет использоваться с видео потоком от веб-камеры с 15-20 fps.

Учитывая, что модель будет работать с веб-камерой на 15-20 fps, а в наборе данных fps варьируется в пределах 27, нужно привести данные к единой частоте кадров.

Возможные подходы:

Приведение всех данных к 15-20 fps (под веб-камеру)

Лучший вариант, так как совпадает с будущими условиями использования.

Если видео имеет больше кадров, чем нужно (например, 27 или 50 fps), сэмплировать его, выбирая каждый n-ый кадр:

n=исходный fpsцелевой fpsn = \frac{\text{исходный fps}}{\text{целевой fps}}n=целевой fpsисходный fps​

Например, если у нас 27 fps, а мы хотим 15 fps, то берём каждый 27/15 ≈ 1.8-й кадр (то есть примерно каждый 2-й кадр).

Если видео имеет меньше кадров, чем нужно (например, 5 fps, а нужно 15), интерполировать недостающие кадры:

* + Линейная интерполяция поз (простая, но может дать резкие скачки).
  + Spline-интерполяция (более гладкие движения).
  + Interpolation через нейронные сети (например, DAIN).

Нормализация временной длины последовательностей

Если последовательности имеют разную длину (из-за разного fps и длительности видео), можно:

* Обрезать до фиксированного количества кадров (например, 30-60 кадров).
* Дополнять нулями или зеркально отраженными кадрами.
* Растягивать/сжимать последовательность (Dynamic Time Warping).

Альтернативный вариант — обучить модель с разным fps

Если ты хочешь, чтобы модель была устойчивой к разным fps, можно оставить данные как есть, но:

* Использовать аугментацию fps во время обучения.
* Добавить в модель позиционное кодирование fps (как в некоторых Transformer-архитектурах).
  1. **Анализ распределения классов.**

Данное распределение похоже на распределение типа "длинного хвоста" (long-tail distribution).

Характерные особенности:

* Есть небольшое количество классов с большим числом примеров (слева).
* Большинство классов имеет относительно мало примеров (справа).
* Плавное убывание количества объектов от популярных к редким классам.

Такое распределение часто встречается в задачах классификации действий, особенно если есть редкие и специфические движения. Это может привести к проблемам при обучении модели, так как редкие классы могут быть плохо распознаны.

Возможные способы обработки:

1. Oversampling (увеличение данных для редких классов)
   * Аугментация данных (например, добавление шума, повороты, изменение скорости выполнения действия).
   * SMOTE или его аналоги для синтетического увеличения выборки.
2. Undersampling (уменьшение данных для частых классов)
   * Искусственное уменьшение выборки в классах с большим числом объектов.
   * Можно комбинировать с oversampling.
3. Взвешивание классов при обучении
   * Использование взвешенной функции потерь (например, CrossEntropyLoss с weight).
   * Focal Loss, если редкие классы особенно важны.
4. Использование методов обучения с учетом длинного хвоста
   * Class-balanced loss (методы типа CB Loss).
   * Meta-Learning подходы для обработки редких классов.
   1. **Проверка на пропущенные значения и шум**

Есть точки, которые выходят за пределы кадра(превышающие 1 по x и y). Около 16 % кадров имеют больше половины пропусков и около половины кадров имеют хотя бы один пропуск.

Изображение выглядит как текст, снимок экрана, График, дисплей

Контент, сгенерированный ИИ, может содержать ошибки.