# Human Pose Estimation with Simple Baseline and Pose-Based Action Recognition

Nicola Bazzani
nicola.bazzani@studenti.unipd.it

Sara Pangrazio
sara.pangrazio@studenti.unipd.it

## Abstract

*Human pose estimation has seen significant progress in recent years, often accompanied by increasing model and system complexity. While such approaches achieve strong performance, they make analysis and comparison more difficult. In this work, we reproduce and adapt a simple baseline model inspired by the approach proposed in [15] for human pose estimation, and evaluate its effectiveness on still images. The model is based on a ResNet backbone with a lightweight deconvolutional head and predicts joint heatmaps. We further show how the estimated poses can be exploited for a pose-based action recognition task through unsupervised pseudo-labeling. Our results highlight the value of simple and reproducible baselines for both pose estimation and higher-level reasoning. The code developed for this project is available at:* `https://github.com/nicolabazzani/VCS-project`

## 1. Introduction

In this project we address the problem of human pose estimation from still images through the implementation of a complete and reproducible pipeline. Starting from raw RGB images, our code predicts the 2D location of human body keypoints and evaluates their accuracy using standard pose estimation metrics. The final objective is to understand how much can be achieved with a simple and well-structured implementation.

The core of our pipeline is a heatmap-based pose estimation model. We use a ResNet backbone pre-trained on ImageNet to extract visual features, and we attach a lightweight deconvolutional head that progressively increases spatial resolution and outputs one heatmap per joint. Each heatmap encodes the probability of a joint being present at a specific pixel location. The network is trained using a mean squared error loss between predicted and ground-truth heatmaps. Before being fed to the model, images are cropped and resized to a fixed resolution using an affine transformation computed from the ground-truth joint bounding box. Data augmentation is applied through random scaling, rotation and horizontal flipping to improve robustness and generalization.

Training is monitored using a simplified version of the Object Key point Similarity (OKS) metric, which provides a continuous measure of the quality of joint localization and is used to select the best model checkpoint. Final performance is evaluated on a held-out test set using the Percentage of Correct Joints (PCJ) and Percentage of Correct Parts (PCP) metrics. PCJ measures whether each predicted joint lies within a scale-normalized distance from the ground truth, while PCP evaluates the correctness of entire body parts by checking whether both endpoints are accurately localized. On the test set, our model achieves a $\text{PCJ}_{0.05}$ of 75.72% and a $\text{PCP}_{0.05}$ of 65.49%.

Once the pose estimation stage is completed, the predicted joint coordinates are used as the basis for an action recognition task. From the joints, our code constructs pose-based feature vectors that describe body geometry and motion-related cues. Since the LSP dataset does not provide action annotations, we first generate pseudo-labels by applying K-means clustering to these features. A systematic exploration of different feature representations and numbers of clusters is performed, and the final clustering configuration is selected based on stability and cluster quality criteria. These pseudo-labels allow us to transform the problem into a supervised classification task.

Using the same pose-derived features, we then train and evaluate several classifiers, including neural models (MLPs) and classical machine learning method such as Random Forests. In addition, we also test the same classification pipeline using a set of manually annotated action labels.

## 2. Related Work

Human Pose Estimation has progressed rapidly with deep learning, reaching strong performance of established benchmarks like MPII [3] and COCO [11]. As performance improved, research moves toward more challenging settings such as multi-person pose estimation and pose tracking [2, 15].At the same time, many high-performing architectures on MPII [5, 7, 13, 16], or COCO [4, 6, 8, 12, 14] benchmark became increasingly more elaborate, compli-

cating ablation and fair comparison. *Simple Baselines for Human Pose Estimation and Tracking* explores how competitive a minimal, standard pipeline can be as a reference model, and we adopt this architecture as our pose estimation model.

## 3. The Dataset

For our analysis we used the Leeds Sports Pose Dataset [10]. It contains 2000 images of mostly people playing sports gathered from Flickr. This dataset included the plain images, the images with poses visualized and the file `joints.mat`, containing the 14 joints locations for each image, along with a binary value specifying joint visibility.

In this dataset only one person's pose per image was estimated, so we followed the same approach in our analysis. The images were all of different sizes, but they are scaled so that the most prominent person would be roughly 150 pixels in length. Figure 1 shows some examples.

To get images of the same size and center them around the person whose pose we needed to estimate, all images were cropped and resized to a fixed resolution of 192×256 using an affine transformation based on the ground-truth joint bounding box.

Because of the limited number of images, data augmentation is necessary. It includes scale $\pm 30\%$, rotation $\pm 40$ degrees and flip.

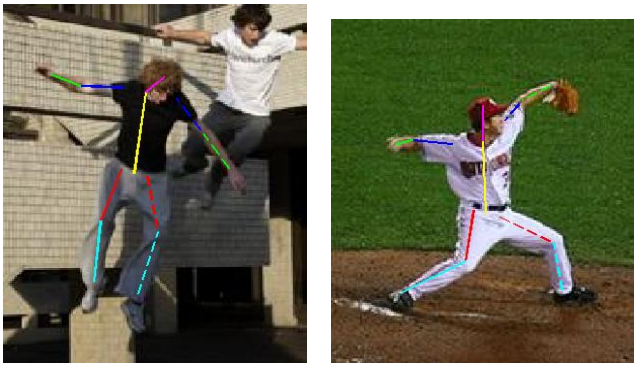

Figure 1: Images with visualized pose number 8 and 16 of LSP dataset

## 4. The Method

### 4.1. Pose Estimation

The goal of this section was to implement a Convolutional Neural Network capable of detecting the 14 joints in the human body.

We followed the approach described in [15]. The advantages of this implementation are the balance between simplicity and performance: even with a relatively simple baseline, it achieves strong accuracy on the pose estimation task.

The pose estimation pipeline is based on a backbone network, ResNet [9] augmented with some deconvolutional layers. The model [15] was trained on COCO human pose benchmark [1], [11] and evaluated on COCO test-dev split, achieving a 73.3 mAP.

Following this procedure, we used a ResNet backbone pre-trained on ImageNet, which is a standard choice for image feature extraction. Then, over the last convolutional stage in ResNet, three deconvolutional layers were added. Each layer had 256 filters with $4 \times 4$ kernel and stride 2. A final $1 \times 1$ convolutional layers outputs the predicted heatmaps.

Heatmaps are central to this approach: the idea is that for every input image, the output produces one heatmap per joint where each pixel represents the probability that the joint is located at that position. Target heatmaps are generated by applying a 2D gaussian kernel centered on the ground-truth joint's coordinates.

The Mean Squared Error (MSE) function was used as loss between the predicted and target heatmaps.

### 4.2. Action Recognition

Our dataset does not include ground-truth action labels. For this reason, we first explored an unsupervised strategy to discover recurring action patterns from joints coordinates, and then introduced manual annotations to train and evaluate supervised classifiers.

#### 4.2.1 Unsupervised Approach

We adopted a two-stage pipeline. First, we used our pose estimation model to extract joint coordinates for each image and built a pose-based feature vector. Then, a clustering algorithm was implemented to separate our dataset into action-like groups. These clusters were treated as *pseudo-labels*, allowing us to train and evaluate different classification models that predict the cluster from the extracted pose features.

KMeans was implemented as clustering algorithm and the number of clusters was set at $K = 3$. This choice was justified by an analysis of cluster quality metrics, accompanied by a manual inspection of samples from each clusters. Stability was measured using Adjusted Rand Index (ARI) across different random initializations, to indicate how reproducible a cluster is. To evaluate cluster separation we used Silhouette, a label-free metric that checks for each sample whether it's closer to its own cluster rather than to the nearest other cluster, and then averages the results. The percentage of the largest cluster was also taken into consideration to avoid degenerate solutions in which one cluster absorbs most samples.

The 2000 initial images were split into three clusters of

1339, 110 and 491 samples. After a manual inspection, we found that the first cluster (cluster 0) captured mainly still or low-motion poses, the second (cluster 1) jumping/acrobatic poses and the third (cluster 2) dynamic, high-motion poses. Some images of the respective clusters are reported in Figure 2.

After generating the *pseudo labels*, a few different classification models were trained to predict the clusters for each image.


Figure 2: Sample results: we reported three images for each cluster.

#### 4.2.2 Supervised Approach

For the supervised setting, we manually labeled the dataset into three different groups: group 0 (313 images) containing jump-affine and acrobatic actions, group 1 (962 images) , containing low motion actions, and group 2 (725 images), containing high motion and dynamic actions. We then trained and evaluated the same set of classification models to predict these manually assigned classes.

#### 4.2.3 Classification Models

The estimated joints coordinates from the Pose Estimation model were used as input for all the classifiers.

- **MLP Baseline:** we first implemented a fully connected MLP that takes as input 2D joint coordinates and outputs 3-class predictions. Before classification, joints are normalized (hip centering and scale normalization) and flattened to vectors of length 28, which is the input size. The network consists of 3 linear layers with ReLU activations and dropout. Training is performed with Adam Optimizer and Cross Entropy loss.
- **Weighted MLP:** this model is equal to our baseline with the exception of the standard cross entropy loss, replacing it with a class-weighted version to address class imbalance. Class weights are computed using inverse class frequency.
- **Deep MLP:** we also evaluated a deeper MLP with the same input and normalization as the baseline. It uses three hidden layers and wider fully connected blocks and stronger regularization. We trained it with Adam Optimizer and Weighted Cross Entropy loss.
- **Random Forest:** finally, we trained a Random Forest classifier with 600 trees. To handle class imbalance we adopted the same weights used for the other models. Before being used as input, the joints were normalized and flattened.

## 5. Experiments

### 5.1. Pose Estimation

The data was split so that $70\%$ would be the training set, $10\%$ the validation set and $20\%$ the test set.

**Training:** during training, performance was monitored using a simplified version of the Object Keypoint Similarity (OKS) metric.

Let $p_{b,j} \in \mathbb{R}^2$ and $g_{b,j} \in \mathbb{R}^2$ be the predicted and ground-truth coordinates of joint $j$ for sample image $b$. The squared error is $d^2_{b,j} = ||p_{b,j} - g_{b,j}||^2_2$. The person scale was approximated by the area $A_b$ of the bounding box enclosing the ground-truth joints. The value $\sigma$ represents the per-joint tolerance and was fixed at $\sigma = 0.05$. Finally, the per-joint similarity was computed as

$$OKS_{b,j} = \exp\Big(-\frac{d^2_{b,j}}{2A_b(2\sigma)^2}\Big) \tag{1}$$

and the reported score was the mean over all joints and samples

$$OKS = \frac{1}{B \cdot J}\sum_{b=1}^{B}\sum_{j=1}^{J} OKS_{b,j} \tag{2}$$

This simplified OKS differs from the official COCO OKS/AP protocol, which uses per-joint tolerances and visibility flags and reports AP across multiple OKS thresholds.

The model was trained for 30 epochs with batch size 64, using Adam Optimizer with learning rate 1e-4. The best checkpoint was selected as the one achieving the highest validation OKS.

**Testing:** the performance on the test set was evaluated using metrics Percentage of Correct Joints (PCJ) and Percentage of Correct Parts (PCP).

A predicted joint $p_{b,j}$ was considered correct when its Euclidean distance to the ground-truth joint is below a fraction of the person scale:

$$||p_{b,j} - g_{b,j}||_2 \leq \alpha \cdot L_b \tag{3}$$

where $L_b$ is the diagonal length of the bounding box , and serves as a proxy for person size.

$PCJ_\alpha$ is computed as the fraction of correctly localized joints under threshold $\alpha$, averaged over joints and samples. $PCP_\alpha$ measures the fraction of correctly localized body parts, where a body part is considered correct if both its endpoint joints are.

Table 1 shows the results we obtained, applying the metrics first to all joints and then only to the visible ones. Figure 3 shows some images of predicted and ground-truth poses.

Table 1: Pose Estimation Results

| $\alpha$ | PCJ | PCP | PCJ vis | PCP vis |
|---|---|---|---|---|
| $\alpha = 0.04$ | 70.17% | 58.19% | 73.66% | 63.35% |
| $\alpha = 0.05$ | 75.72% | 65.49% | 79.11% | 70.66% |
| $\alpha = 0.06$ | 80.48% | 72.54% | 83.70% | 77.58% |
| $\alpha = 0.07$ | 82.76% | 75.29% | 85.65% | 80.20% |
| $\alpha = 0.08$ | 84.36% | 77.25% | 87.10% | 82.06% |
| $\alpha = 0.09$ | 85.54% | 78.68% | 88.12% | 83.41% |
| $\alpha = 0.1$ | 86.64% | 80.13% | 88.94% | 84.34% |

## 5.2. Action Recognition

### 5.2.1 Unsupervised Approach

For the KMeans stage, we tuned both the number of clusters $K$ and feature representations via grid search. We fit the different models on the training set and evaluated cluster-quality on the validation set.

We tried $K \in \{2, ..., 10\}$ over three different feature representations:


Figure 3: Sample results: example of estimated poses on the left and ground-truth poses on the right

- Method (A) : features designed to emphasize jump-like actions, like knee bend angles, feet separation, average ankle height and if a person was upside-down or elongated.
- Method (B) : features based on the main angles between body parts, like knees, elbows, hips, shoulders.
- Method (C) : a concatenation of (A) and (B).

For each configuration, we ran multiple random seeds and monitored ARI mean and standard deviation. We also tracked Silhouette score and the percentage of samples in the largest cluster.

Results are reported in Table 2.

For Method A, $K = 2$ achieves the highest ARI and Silhouette, but produces an unbalanced partition, with larges cluster containing 90% of the samples, which indicates an almost trivial split. $K = 3$ remains highly stable among various seeds while improving balance (largest cluster at 69%). Moreover, with $K = 3$, Method A outperforms Methods B and C in both stability and separation. For this reason, we chose Method A with $K = 3$ for generating pseudo-labels.

**Classification Models Training:** during training, the PoseNet model is kept in evaluation mode so only the action classifiers parameters are updated. PoseNet heatmaps are converted to 2D joints coordinates. We trained Adam Optimizer with 1e-3 learning rate for 20 epochs.

During the training of RandomForest Classifier, we evaluated performance on validation set using accuracy and F1-macro score.

**Testing:** final evaluation is performed on the test set and reported using per-class error rate, defined as 1 - recall, and per-class F1 score. The results are summarized in Table 3 and Figure 4.

Table 2: KMeans grid search summary across feature representations.

(a) Feature method A

| $K$ | ARI (mean±std) | Silhouette | Largest cluster (%) |
|---|---|---|---|
| 2 | 1.000±0.000 | 0.535 | 91.50% |
| **3** | **0.734±0.337** | **0.331** | **69.60%** |
| 4 | 0.520±0.287 | 0.283 | 59.15% |
| 5 | 0.582±0.214 | 0.217 | 50.85% |
| 6 | 0.626±0.206 | 0.231 | 50.20% |
| 7 | 0.774±0.117 | 0.235 | 49.55% |
| 8 | 0.779±0.110 | 0.224 | 45.95% |
| 9 | 0.844±0.065 | 0.226 | 45.15% |
| 10 | 0.825±0.078 | 0.224 | 44.25% |

(b) Feature method B

| $K$ | ARI (mean±std) | Silhouette | Largest cluster (%) |
|---|---|---|---|
| 2 | 1.000±0.000 | 0.341 | 83.50% |
| 3 | 0.962±0.027 | 0.118 | 53.00% |
| 4 | 0.785±0.233 | 0.114 | 47.25% |
| 5 | 0.503±0.188 | 0.105 | 38.60% |
| 6 | 0.462±0.110 | 0.103 | 33.85% |
| 7 | 0.492±0.127 | 0.103 | 29.60% |
| 8 | 0.491±0.107 | 0.097 | 25.15% |
| 9 | 0.469±0.097 | 0.094 | 23.60% |
| 10 | 0.459±0.113 | 0.090 | 21.40% |

(c) Feature method C

| $K$ | ARI (mean±std) | Silhouette | Largest cluster (%) |
|---|---|---|---|
| 2 | 1.000±0.000 | 0.417 | 89.50% |
| 3 | 0.601±0.408 | 0.146 | 59.55% |
| 4 | 0.802±0.160 | 0.108 | 46.75% |
| 5 | 0.663±0.199 | 0.107 | 40.70% |
| 6 | 0.610±0.113 | 0.112 | 36.50% |
| 7 | 0.605±0.127 | 0.116 | 32.80% |
| 8 | 0.577±0.096 | 0.113 | 29.30% |
| 9 | 0.616±0.111 | 0.110 | 27.55% |
| 10 | 0.574±0.112 | 0.108 | 26.20% |

The baseline MLP performs well on class 0–1 but fails on class 2 (66% error) . Introducing class weights substantially improves class 2 performance, and the deep weighted MLP provides the best overall balance among the neural models, obtaining a mean F1 score of 78%. The Random Forest achieves the highest F1 on class 0 (88%), but remains weaker, in comparison to the other models, on class 1 and class 2.

Table 3: Summary Tables of Unsupervised Approach

| Model | Class | Error Rate | F1 Score |
|---|---|---|---|
| Baseline MLP | class 0 | 3.73% | 87.0% |
| | class 1 | 23.08% | 81.16% |
| | class 2 | 66.04% | 45.6% |
| Weighted MLP | class 0 | 13.43% | 86.2% |
| | class 1 | 11.54% | 83.6% |
| | class 2 | 40.57% | 60.9% |
| Deep MLP | class 0 | 12.31% | 86.9% |
| | class 1 | 11.54% | 86.8% |
| | class 2 | 38.68% | 63.1% |
| Random Forest | class 0 | 4.48% | 88.3% |
| | class 1 | 26.92% | 77.6% |
| | class 2 | 54.72% | 56.1% |

| Model | Mean Error Rate | Mean F1 Score |
|---|---|---|
| baseline MLP | 30.95% | 71.3% |
| Weighted MLP | 21.84% | 76.9% |
| Deep MLP | 20.84% | 78.9% |
| RandomForest | 28.71% | 74.0% |


Figure 4: Unsupervised Approach: error rate obtained with different models.

### 5.2.2 Supervised Approach

Since the loss function was decreasing more slowly than in the previous case, the neural models were trained over 30 epochs. Besides the different class weights, everything else was kept the same.

Results can be visualized in Table 4 and Figure 5. The baseline MLP shows uneven performance, with a large class error on class 2 (54% error). Weighted MLP improves class

2 error (40% error), but worsens class 1 performance. The DeepMLP provides the best overall balance. The RandomForest achieves the solid F1 scores on classes 1 and 2, but remains weaker on class 0 compared to other models.

Table 4: Summary Tables of Supervised Approach

| Model | Class | Error Rate | F1 Score |
|---|---|---|---|
| Baseline MLP | class 0 | 27.45% | 69.8% |
| | class 1 | 20.51% | 72.9% |
| | class 2 | 54.55% | 52.0% |
| Weighted MLP | class 0 | 13.73% | 65.2% |
| | class 1 | 40.51% | 65.7% |
| | class 2 | 40.26% | 59.0% |
| Deep MLP | class 0 | 13.73% | 68.2% |
| | class 1 | 25.64% | 74.4% |
| | class 2 | 40.91% | 64.8% |
| Random Forest | class 0 | 29.41% | 66.1% |
| | class 1 | 24.10% | 72.7% |
| | class 2 | 46.10% | 58.5% |

| Model | Mean Error Rate | Mean F1 Score |
|---|---|---|
| baseline MLP | 34.31% | 64.9% |
| Weighted MLP | 31.5% | 63.3% |
| Deep MLP | 26.76% | 69.1% |
| RandomForest | 33.20% | 65.8% |

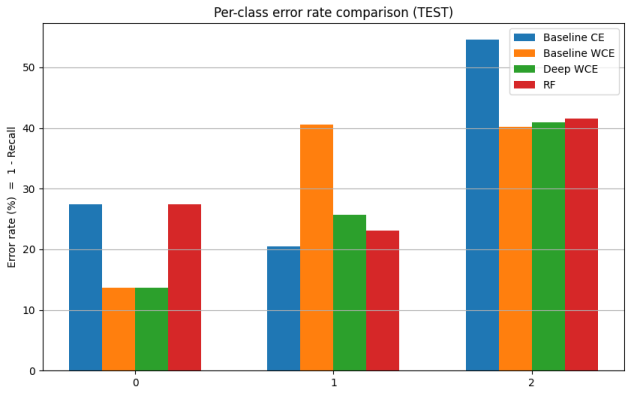


Figure 5: Supervised Approach: error rate obtained with different models.

## 6. Conclusions

In this project we showed that strong results in human pose estimation can be obtained even with a simple and well-structured baseline. A heatmap-based model with a ResNet backbone and a lightweight deconvolutional head is sufficient to achieve accurate localization of body joints on still images, as confirmed by the PCJ and PCP metrics. This demonstrates that high architectural complexity is not strictly necessary and that simple baselines can already provide reliable performance.

For the action recognition task, we observed that working with single images makes the problem more challenging, since actions are inherently dynamic and temporal information is missing. However, by using the estimated joints and constructing pose-based features based on relative joint positions, distances between body parts and joint angles, we were still able to obtain meaningful results for general motion categories such as still, low-motion and dynamic movements. This shows that pose information alone already enables a form of basic action discrimination.

As future work, a natural extension would be to move to a temporal setting by analyzing consecutive temporally ordered images, which would allow the exploitation of motion information and lead to more accurate and specific action recognition. Another important direction would be to extend the pipeline to handle multiple people in the same image, estimating and analyzing several poses simultaneously. Finally, making the system robust to partial visibility, where people are not fully contained within the image, would further increase its applicability in real-world scenarios.

## References


[1] Coco leaderboard. `http://cocodataset.org`. Accessed: 2026-01-20.

[2] Mykhaylo Andriluka, Umar Iqbal, Eldar Insafutdinov, Leonid Pishchulin, Anton Milan, Juergen Gall, and Bernt Schiele. Posetrack: A benchmark for human pose estimation and tracking, 2018.

[3] Mykhaylo Andriluka, Leonid Pishchulin, Peter Gehler, and Bernt Schiele. 2d human pose estimation: New benchmark and state of the art analysis. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3686–3693, 2014.

[4] Zhe Cao, Tomas Simon, Shih-En Wei, and Yaser Sheikh. Realtime multi-person 2d pose estimation using part affinity fields, 2017.

[5] Yu Chen, Chunhua Shen, Xiu-Shen Wei, Lingqiao Liu, and Jian Yang. Adversarial posenet: A structure-aware convolutional network for human pose estimation, 2017.

[6] Yilun Chen, Zhicheng Wang, Yuxiang Peng, Zhiqiang Zhang, Gang Yu, and Jian Sun. Cascaded pyramid network for multi-person pose estimation, 2018.

[7] Xiao Chu, Wei Yang, Wanli Ouyang, Cheng Ma, Alan L. Yuille, and Xiaogang Wang. Multi-context attention for human pose estimation, 2017.

[8] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn, 2018.

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015.

[10] Sam Johnson and Mark Everingham. Clustered pose and nonlinear appearance models for human pose estimation. In *Proceedings of the British Machine Vision Conference*, 2010. doi:10.5244/C.24.12.

[11] Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollár. Microsoft coco: Common objects in context. In *European Conference on Computer Vision (ECCV)*, 2014.

[12] Alejandro Newell, Zhiao Huang, and Jia Deng. Associative embedding: End-to-end learning for joint detection and grouping, 2017.

[13] Alejandro Newell, Kaiyu Yang, and Jia Deng. Stacked hourglass networks for human pose estimation, 2016.

[14] George Papandreou, Tyler Zhu, Nori Kanazawa, Alexander Toshev, Jonathan Tompson, Chris Bregler, and Kevin Murphy. Towards accurate multi-person pose estimation in the wild, 2017.

[15] Bin Xiao, Haiping Wu, and Yichen Wei. Simple baselines for human pose estimation and tracking, 2018.

[16] Wei Yang, Shuang Li, Wanli Ouyang, Hongsheng Li, and Xiaogang Wang. Learning feature pyramids for human pose estimation, 2017.