Human Pose Estimation and K-mean Clustering

By Cheuk-Hang Tse

# Abstract

This write-up describes the implementation process of performing human pose estimation and k-mean clustering. OpenCV is used in Human Pose Estimation to read the Deep Neural Network and perform localization on input image to find human pose key point locations. K-mean clustering is used to read dataset and train the clustering model for clustering new human pose key point locations to similar image.

# Introduction

Human Pose Estimation was used to locate joint of a human within an image. However, there is more information to discover from the joint locations, such as pose clustering. By clustering joint locations to similar locations, the model can find images that have a similar human pose to the input image. In this write-up, human pose estimation and k-mean clustering are used to find joint locations of an image and find similar images based on the joint locations.

# Human Pose Estimation

## Key point Table

The Human Pose Estimation model in this write-up uses MPII Human Pose Estimation as training data. The model is trained with 15 points:

|  |  |
| --- | --- |
| **Key Point Name** | **Index** |
| Head | 0 |
| Neck | 1 |
| Right Shoulder | 2 |
| Right Elbow | 3 |
| Right Wrist | 4 |
| Left Shoulder | 5 |
| Left Elbow | 6 |
| Left Wrist | 7 |
| Right Hip | 8 |
| Right Knee | 9 |
| Right Ankle | 10 |
| Left hip | 11 |
| Left Knee | 12 |
| Left Ankle | 13 |
| Chest | 14 |

**Table 1: Corresponding Index per Each Key Point**

Each key point is connected to one or more key points which form a skeleton. The image below shows the key point location of a person working out:

A picture containing wall, indoor

Description automatically generated

**Figur­­e 1: Key point location of a person working out and connection for each key point location**

## Deep Neural Network

The deep neural network model is designed and trained by the Perceptual Computing Lab at Carnegie Mellon University. The model takes a colored image of a size of w and h. Then produce a vector of 2D location of the key points of a person in the image. The model used in this project is subjected to single human pose estimation. The model architecture is as follows:

Diagram

Description automatically generated

**Figure 2: Multi-Person Human Pose Estimation model architecture**

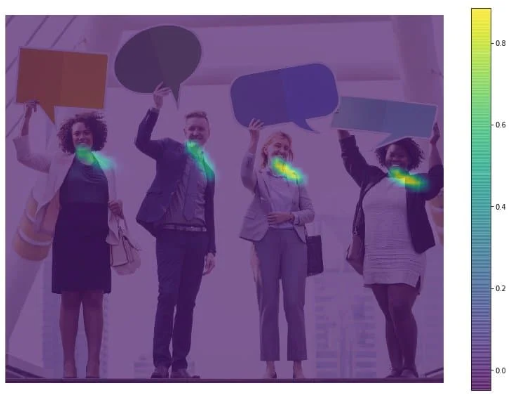
The model detects a human pose in 3 stages. First, 10 layers of the Very Deep Convolutional Networks (VGGNet) are used to create a feature map for the input image.Second, a 2-branch multistage Convolution Neural network (CNN) is used. The first branch is used to predict a set of confidence map of body parts location. The following figure shows an example of the confidence map and its affinity map for detecting left shoulder in an image.

A group of people holding hats

Description automatically generated with medium confidence

**Figure 3: Confidence map for Left Shoulder with corresponding affinity map on the left**

Then, the second branch is used to predict a set of 2D vectors of key point affinities and encode the degree of association between key points. The figure below shows the key point affinity between Neck and Left Shoulder:



**Figure 4: Key Point Affinity maps for the Neck and Left Shoulder pair**

The last stage is to parse the confidence and affinity maps together into a greedy algorithm. The algorithm will produce the 2D key points location for the people inside the image.

# K-mean Clustering

The next stage after finding 2D key points of a human pose is to perform k-mean clustering to find image that have people with similar pose. The clustering class takes the number of clusters as a parameter for training the model. If no specific dataset file name is inputted to the clustering class, the class will use the default dataset, “test.csv”.

## Read and Save Dataset

There is a specific formatting for storing key point in the dataset. The first column must be the file name of the image that produce the corresponding key points in the upcoming columns. Every 2 columns represent a key point’s coordinate, starts from 0 key point index to the 14 key point index. Each row of the dataset will be saved inside a structure called Cluster\_Point, which store information, such as coordinates, cluster index, file name, of a point in the clustering model. When saving Cluster\_Point into a dataset, the class will first store the file name, then corresponding key points’ coordinate.

## Train Clustering Model

After reading the dataset, the cluster model trains clusters based on the inputted number of clusters. First, the class will initialize the clusters on random location. Then, the model will train the clusters by 100 iterations of the following steps:

1. Assign clusters to Cluster\_Point based on the Euclidean distance with the cluster.
2. Find the sum of coordinates that share the same cluster.
3. Calculate the average of the sum of coordinates, which will be the new cluster’s centroid.
4. Reset the minimal distance for each Cluster\_Point.

By the end of the iterations, the cluster coordinates are finalized and ready for clustering new key points

## Clustering Key Points

To cluster new key points, the class iterates the list of clusters and find the closest cluster, by Euclidean distance, to the inputted key points. Then, it iterates the list of Cluster\_Point and find points that share the same cluster index, then save the file name of the point in a list. At last, return the list of file name that shares similar human pose.

# Result

The batch file must input 3 parameters: device (“gpu” or “cpu”), inputFile (image file name), k (number of clusters).

start "" ../x64/Debug/CheukHangTse\_Project.exe cpu 000033016.jpg 15



**Figure 5: 000033016.jpf image file in the result section**

The image file is inputted into the human pose estimation deep neural network and produce the 2D key points. After adding circle on key point coordinate and straight-line connections between key points, the following image is saved to the disk.

A picture containing indoor, wall, floor, window

Description automatically generated

**Figure 6: Output-Skeleton.jpg image file that draw key points location and connections**

The key points location is then used to find image with similar human pose key point locations. To fit the clustering model, the key points location must be pre-processed. The key points are normalized by using the min-max normalization method.

After normalization, key points are passed to the model to find image with similar pose. There are 13 related images displayed. One of the results is the following image:



**Figure 7: single.jpeg image file that was displayed as a related image to the 000033016.jpf image file**

By running the “single.jpeg” as input file, the code returns the output skeleton image file:



**Figure 8: Output-Skeleton.jpg image file after running single.jpeg as input file**

Comparing the location of the key points between two images, the vertical location of the key points is very similar, which is why these two images are in the same cluster.

# Conclusion

The deep neural network model shows successful pinpointing for single human pose estimation but suffers from multi-human pose estimation. The model provides accurate human pose estimation result for the k-mean clustering model. The K-mean clustering model shows success on clustering similar key points, finding image with similar pose, and providing continuous way to update dataset.

# Future Accomplishments

1. Provide cosine similarity option to find the top x number of image with similar pose
2. Find ways to allow multi-human pose estimate and k-mean clustering