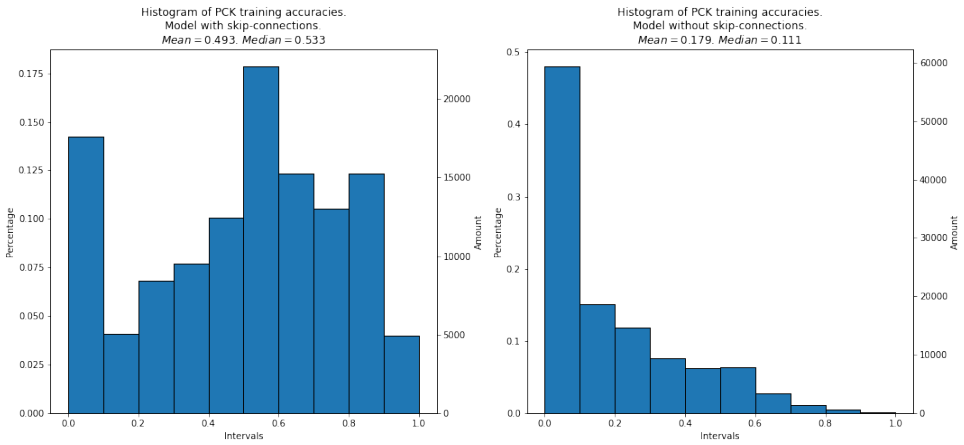


Figure 1: Histogram of PCK training accuracies of the model with the skip-connections enabled (left) and the model with the skip connections disabled (right).

1 Understanding the Model

In the following section we will be interpreting the model developed in section **??**, with the intention of getting an understanding of how the model works.

1.1 Motivation

Deep learning models are often complex and work like a *black boxes*. By that it is meant, that when a network is given some input, the model simply just returns some output without any explanation or reasoning behind the output. This can often be a problem, especially in cases where the output of the network can result in a life or death situation of a human. For that reason, understanding and explaining how a network works can be very important - this is what is called *explainable ai* (*XAI*)

Selvaraju *et al.* [4] argues that there are three cases for using explainable ai:

1. When the network performs worse than humans, an understanding of the network can help us improve the performance of the model.
2. When the network is on par with humans, an understanding of the network is trivial for humans to build trust in the network, as we can understand its strengths and weaknesses
3. When the network performs better than humans, an understanding of the network can teach humans how to perform better.

Throughout section 1 we will be getting an understanding of our model developed in section **??**, as we wish to understand its strengths and weaknesses.

1.2 Verifying the Effects of Skip-Connections

Olsen [3] and Newell [2] claims, that the skip-connections are used in order to recreate details that are lost during the encoder-phase. Throughout subsection 1.2 we will be verifying or refuting the claim of the effect of the skip-connections. To do so we will be using two models based on the same network:

1. The trained Stacked Hourglass from section **??**
2. The trained Stacked Hourglass from section **??**, but with the skip-connections disabled.

Thus, the second model has not been retrained and is identical to the first model, however, without its skip-connections.

In Figure 1 the distributions of the PCK training accuracies of the two models have been visualized. We have decided to make use of the training data for computing the PCK accuracies, as we want to look at the data, that the model has been trained on. By looking at the two distributions we can clearly see how to model without its skip-connections performs much worse, than the model with its skip-connections.

To further understand the decrease of accuracy in the case where the skip-connections are disabled, we have in Figure 2 visualized 20 samples from the training dataset, where the model with skip-connections has an 100% PCK accuracy score. Next to each image the ground truth heatmaps, or the prediction by the model with skip-connections, and the prediction by the model without skip-connections has been visualized.

By looking at Figure 2 we can see, that the model without its skip-connections often struggles with smaller joints, such as the eyes, ears or nose, whereas it performs better, however still not always perfect, on bigger joints, such as the shoulders, hips or knees. This is probably due to the fact, that the details of the smaller joints has a bigger chance of being lost by the max pooling layers in the encoder. Without the skip-connections their information is thus lost, resulting in bad predictions. Thus, we can verify Olsen's [3] and Newell's [2] claims, that the skip-connections are used for recreating details lost in the encoder.

1.3 Shape Analysis of the Latent Space

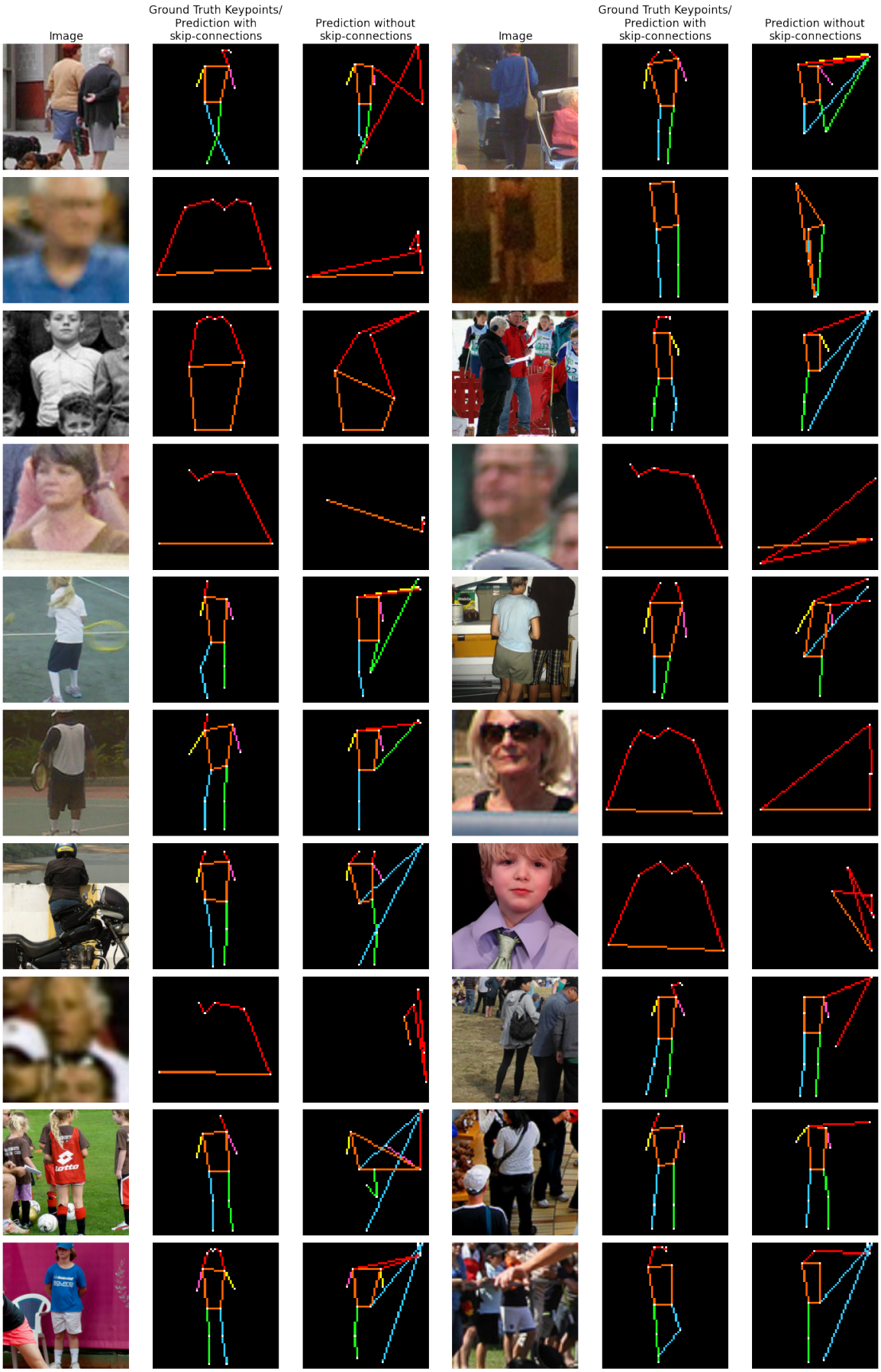


Figure 2: 20 samples of images correctly predicted by the model with skip-connections enabled, the corresponding ground truth heatmap and predictions by the model with skip-connections disabled.

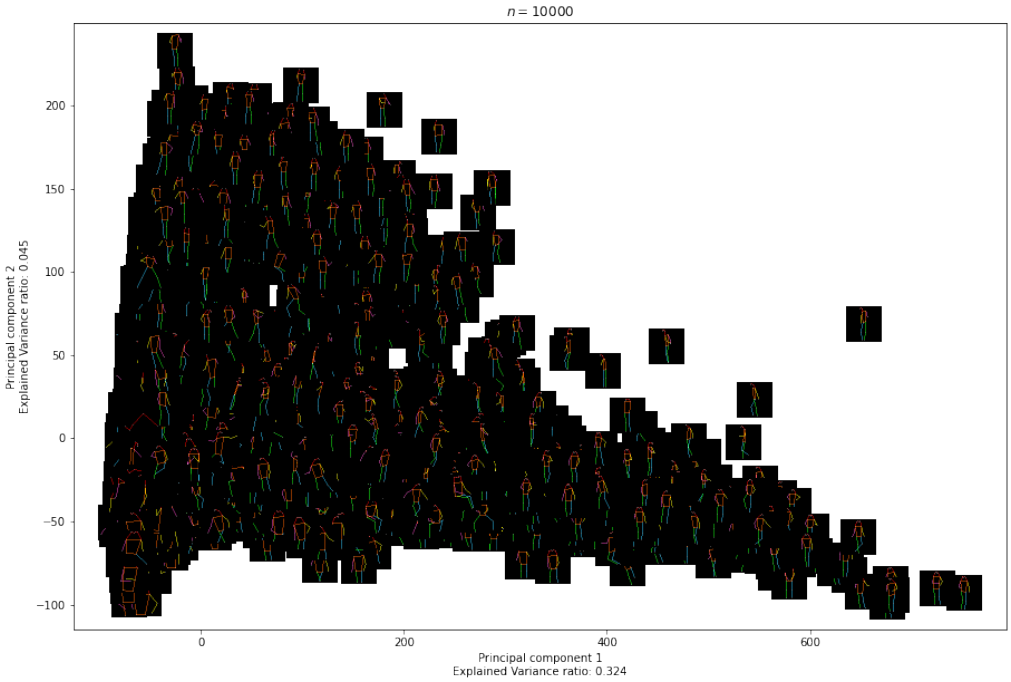


Figure 3: Plot of 10.000 samples of the latent space of the trained model, with the corresponding ground truth heatmaps

1.4 Using Clustering to Seperate the Latent Space

In subsection **??** we described how we decided to use the Stacked Hourglass for the pose estimation, as it is similar to Autoencoders. This makes the model useful for encoding the data into a lower dimension, resulting in the lower dimensional data in the bottleneck of the model a great representation of how the input data actually is. By exploring this representation of the data in the bottleneck, we can get an understanding of how the model relate similar data to each other.

We start off by feeding training data through the encoder of the model and storing the output of the third residual module in the bottleneck. Due to memory constraints only 10.000 random samples were used. We decided to make use of training data for this, as we wish to look at what features of the training data that the model has learned. Each output of the bottleneck is a $4 \times 4 \times 256$ tensor, which we flattened to a 4.096 vector. Each vector was then stacked, forming a 10.000×4.096 matrix of the *latent space*.

If we take this latent space matrix, project it down to 2 dimensions using PCA and visualize the samples with their corresponding ground truth heatmaps, we get the plot visualized in Figure 3. The plot only explains about 37% of the variance of the original data, however, we can clearly see how there is some specific structure in the data, as samples that are somewhat similar are close to each other, however, with a few outliers.

To see how to model seperates the data in the latent space, we will be using K-Means. Choosing the optimal K can often be difficult, as it is often not clear how many clusters there are in the data. For choosing the optimal k the *Silhouette score* is often computed, following the

Algorithm 1 Compute Silhouette Score [1]

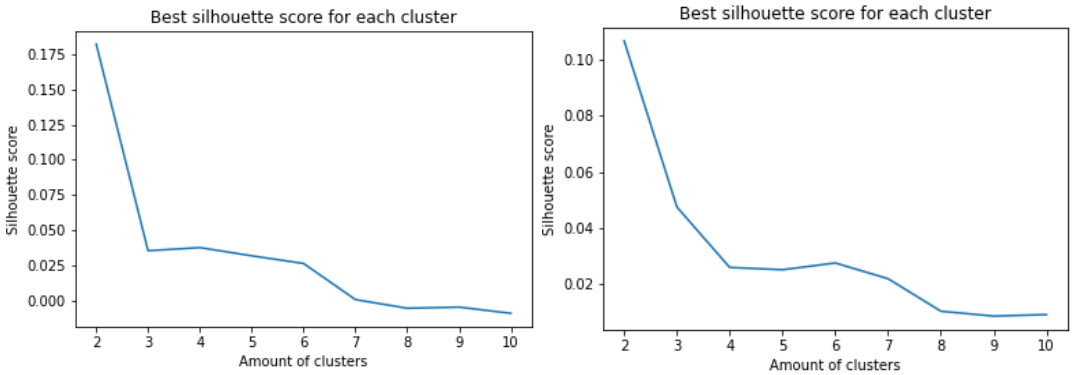
Require: Clusters: $\boldsymbol{C}_0, \boldsymbol{C}_1, ..., \boldsymbol{C}_{k-1}$

for each cluster $\boldsymbol{C}_i$ **do**
  for each sample $\boldsymbol{x} \in \boldsymbol{C}_i$ **do**
    Compute the mean euclidean distance from $\boldsymbol{x}$ to the other samples in the same cluster: $a(\boldsymbol{x}) = \frac{1}{|\boldsymbol{C}_i|-1} \sum_{\boldsymbol{y} \in \boldsymbol{C}_i} D(\boldsymbol{x}, \boldsymbol{y})$
    Compute the mean euclidean distance from $\boldsymbol{x}$ to the nearest other cluster $\boldsymbol{C}_j$: $b(\boldsymbol{x}) = \frac{1}{|\boldsymbol{C}_j|} \sum_{\boldsymbol{z} \in \boldsymbol{C}_j} D(\boldsymbol{x}, \boldsymbol{z})$
    Compute the Silhouette of $\boldsymbol{x}$: $s(\boldsymbol{x}) = \begin{cases} 1 - \frac{a(\boldsymbol{x})}{b(\boldsymbol{x})} & \text{if } a(\boldsymbol{x}) < b(\boldsymbol{x}) \\ 0 & \text{if } a(\boldsymbol{x}) = b(\boldsymbol{x}) \text{ or } |\boldsymbol{C}_i| = 1 \\ \frac{b(\boldsymbol{x})}{a(\boldsymbol{x})} - 1 & \text{if } a(\boldsymbol{x}) > b(\boldsymbol{x}) \end{cases}$
return mean of the Silhouettes

pseudocode visualized in 1. For computing the Silhouette score, various values of K are used for training various K-Means models. After each model has been trained, let a_i be the average distance of the ith sample to the other samples in the same cluster as ith sample. Then, let b_i be the average distance of the ith sample to the samples in the nearest cluster. Ideally, we want $a_i < b_i$, as $b_i < a_i$ means that the ith sample probably has been grouped to the wrong cluster. For that reason, the ith silhouette score is set to $1 - \frac{a_i}{b_i}$ if $a_i < b_i$ or $\frac{b_i}{a_i} - 1$ if $a_i > b_i$. By the end of the algorithm the mean silhouette score is returned. By computing the silhouette score for various values of K, the K with the silhouette score closest to 1 is chosen as the optimal K [1].

When running the K-Means algorithm on the latent space, we use $K = 2, 3, ..., 10$, where the algorithm is retrained 10 times with different initial centroid position for each K. For each run we record the Silhouette score, where the highest Silhouette score for each K has been visualized in Figure 4a. By looking at Figure 4a we can clearly see, how the optimal K for the model is when $K = 2$.

The results of running the K-Means model with $K = 2$ has been visualized in Figure 5. The K-Means model were ran on the latent space in all of the 4.096 dimensions and only each cluster were projected down to 2 dimensions for the purpose of visualization. By looking at Figure 5 we can see how the two clusters has different content: where Cluster 0 focuses more on al-



(a) Latent space consists of 10.000 randomly chosen training samples

(b) Latent space consists of 7.017 fully-annotated training samples.

Figure 4: Silhouette score of running various K-Means models on different data from the latent space.

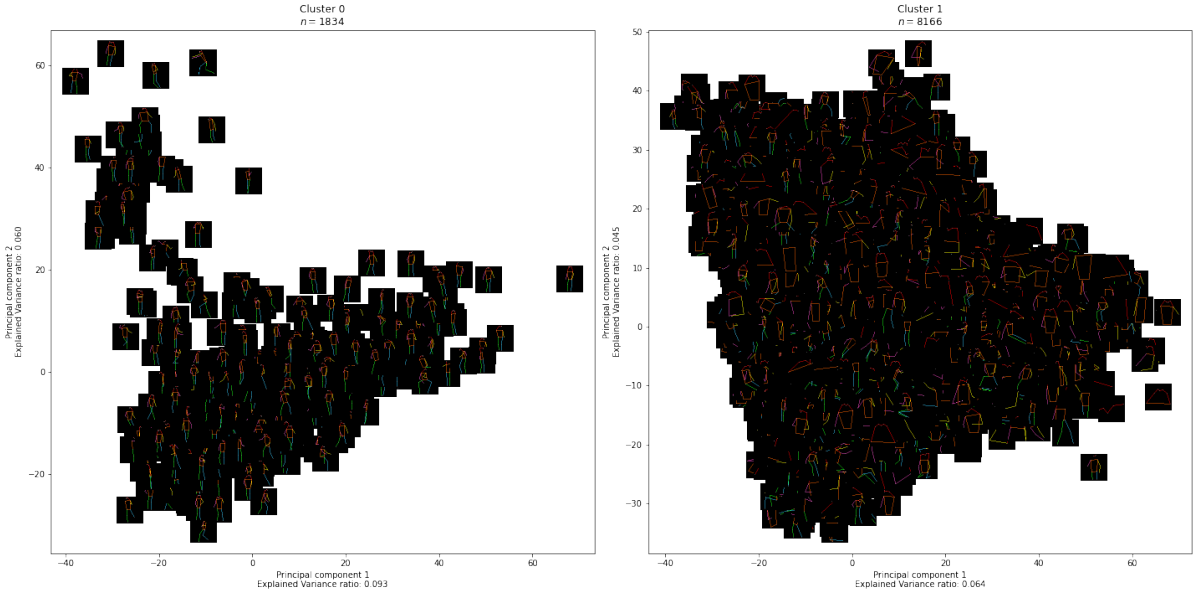


Figure 5: The resulting clusters of running a K-Means model with $K = 2$ on the latent space consisting of 10.000 random training samples

most fully annotated samples, Cluster 1 focuses more on samples that have a lot of keypoints missing. This is also easy to see if we look at the ground truth heatmaps of the samples closest to the centroids of the two clusters, as visualized in Figure 6. By doing so we can see, that the ground truth heatmap of the closest sample to the centroid of Cluster 0 almost has all of its joints annotated, whereas the ground truth heatmap of the closest sample to the centroid of Cluster 1 only consists of 2 keypoints.

Although there are differences in the two clusters in Figure 5, there are still quite a lot of miss-classified samples. To overcome this problem we remove all of the not-fully annotated samples and instead use all of the 7.017 fully-annotated samples of the training set, again fed through the network and outputted by the third residual module in the bottleneck. By doing so we get the Silhouette scores visualized in Figure 4b, where we again clearly see, that $K = 2$ is the optimal value of K.

The two clusters, resulted by only using fully-annotated samples, have been visualized in Figure 7 and the corresponding closest ground truth heatmaps for the samples closest to the centroids have been visualized in 8. Like before, the K-Means model were ran on the data in full dimension to create the two clusters, which then were projected down to 2 dimensions

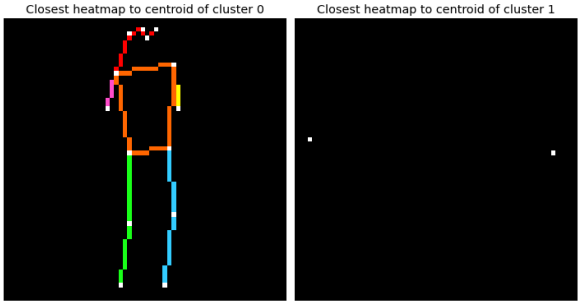


Figure 6: Closest points to the centroids of the two clusters from running K-Means on the latent space consisting of 10.000 random training samples

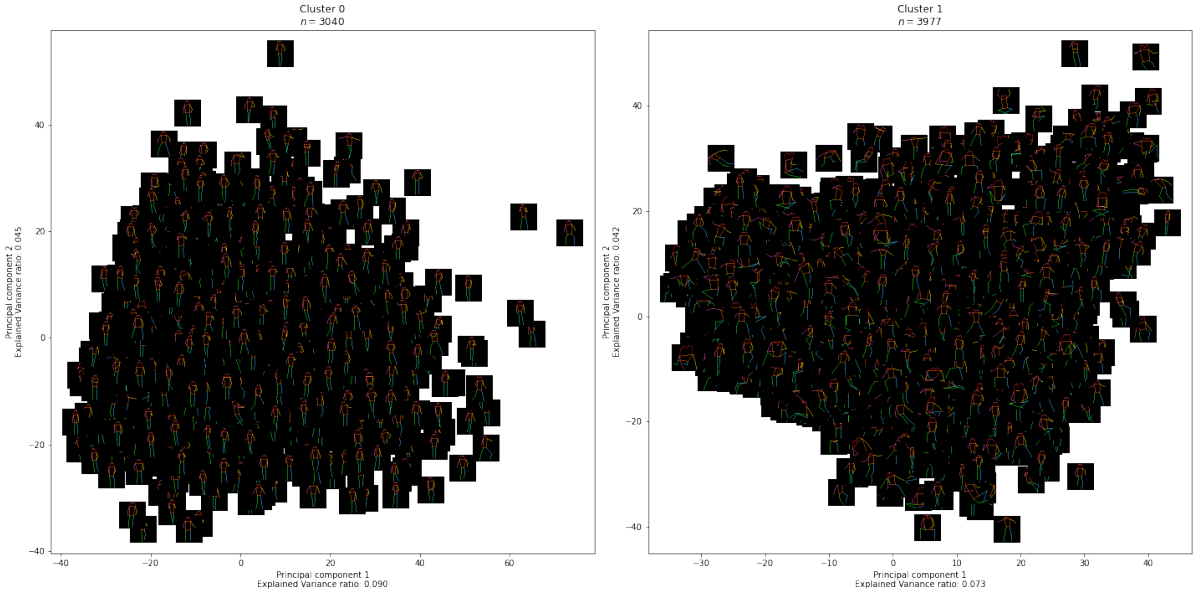


Figure 7: The resulting clusters of running a K-Means model with $K = 2$ on the latent space consisting of 7.017 fully annotated training samples

using PCA for the purpose of visualization. By looking at the figure we clearly see how the content of Cluster 0 contains samples that are stationary, whereas the samples of Cluster 1 carry a lot more movement. This is also the case for the ground truth heatmaps of the samples closest to the centroids, visualized in Figure 8, as we can see, that the heatmap for Cluster 0 is more straighten, whereas the heatmapf for Cluster 1 is more bent and looks like it is in more movement. The two clusters does have a lot less missclassifications, than it was the case with the two clusters in Figure 5. The missclassifications could explain the not-optimal performance of the model as this could mean, that the network has not fully learned the differences between certain positions and where the positions should be placed in the latent space.

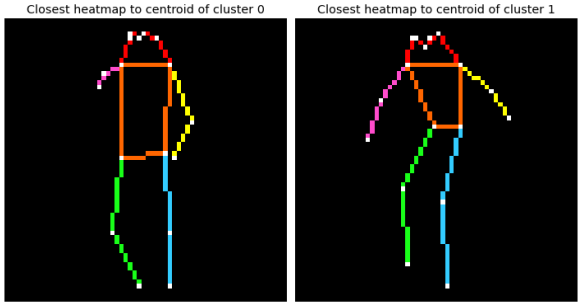


Figure 8: Closest points to the centroids of the two clusters from running K-Means on the latent space consisting of 7.017 fully annotated training samples