Chasing the Tail in Monocular 3D Human Reconstruction with Prototype Memory
Yu Rong, Ziwei Liu and Chen Change Loy

Abstract—Deep neural networks have achieved great progress in single-image 3D human reconstruction. However, existing methods still fall short in predicting rare poses. The reason is that most of the current models perform regression based on a single human prototype, which is similar to common poses while far from the rare poses. In this work, we 1) identify and analyze this learning obstacle and 2) propose a prototype memory-augmented network, PM-Net, that effectively improves performances of predicting rare poses. The core of our framework is a memory module that learns and stores a set of 3D human prototypes capturing local distributions for either common poses or rare poses. With this formulation, the regression starts from a better initialization, which is relatively easier to converge. Extensive experiments on several widely employed datasets demonstrate the proposed framework’s effectiveness compared to other state-of-the-art methods. Notably, our approach significantly improves the models’ performances on rare poses while generating comparable results on other samples.

Index Terms—Motion Capture, 3D Pose Estimation, Clustering

I. INTRODUCTION

Recovering 3D human model from a single monocular image can facilitate numerous applications in augmented reality and creative entertainment. Most existing methods [1], [4], [5], [6], [7] employ a parametric 3D human shape model known as Skinned Multi-Person Linear Model (SMPL) [8] to represent 3D human and use deep convolutional neural networks (DCNN) to estimate its parameters. They also employ 3D human prototype, such as mesh template [4], [7] or mean parameters [1], [5], as the foundation to predict 3D human models. Using these methods, images with common poses are satisfactorily predicted. However, performances of these models decline drastically when applied to images with rare poses or uncommon views, as shown in the third row of Fig. 1 (c).

The performance degradation in predicting rare poses is primarily caused by the usage of a single 3D human prototype for all the samples. The single human prototype calculated from these datasets, as visualized in Fig. 1 (a), is more similar to common poses with usual viewpoints, such as standing. Models trained with this single prototype inherently bias towards the common poses while performing less well on rare poses. To illustrate this phenomenon, we show the visualization of the single human prototype used by SPIN [1] in Fig 1 (a) and calculate the Euclidean distance of each sample to the human prototype. As depicted in Fig. 1 (b), when a single prototype is adopted, two state-of-the-art models, namely SPIN [1] and DecoMR [4] perform well on samples that are close to the human prototype. However, their reconstruction errors rise rapidly on samples that have a larger distance to the mean parameters.

Take SPIN [1] and Human3.6M [2] for instance, the recon-
struction errors of samples with distance to mean parameters of around 4.3 is 20% larger than those with distance to mean parameters of around 2.0. The performance drop is more severe on challenging in-the-wild datasets such as 3DPW \[3\]. We further show several typical examples in Fig. 1 (c). It is observed that current state-of-the-art models’ prediction becomes imprecise when the samples’ poses gradually differ from the human prototype. For better clarification, in the following of the paper, we call the samples that are close to the human prototype head classes and the samples far from the human prototype tail classes. Typically, head classes are composed of common poses while rare poses majorly lie in tail classes.

To ameliorate the models’ performance collapse on tail classes, we propose a prototypical memory network, PM-Net. The core of PM-Net is a memory module that stores multiple 3D human prototypes, each of which covers a non-overlapping subset of the data. Instead of using one single prototype for all data samples, we assign each data to the closest prototype in the memory. In this way, the distance of data to its assigned prototype is significantly reduced. The regression process starts from a much better initialization. Furthermore, the regression process is transformed to be performed on compact local distributions instead of on the global yet sparse distribution. During inference, a classifier is employed to assign input data to the corresponding prototype, which will serve as the basis for SMPL parameter regression.

Previous works such as LCR-Net \[9\] also adopts multiple human prototypes by applying standard K-Means clustering on 3D poses data. Nevertheless, it is not feasible to directly apply the same method in 3D human reconstruction. Firstly, although the distances between 3D poses can be easily measured by Euclidean distance between coordinates, it is not reasonable to use the same metric on SMPL parameters. Distances between vertices are more reasonable. Furthermore, the weights of each vertex should be elaborately selected in calculating the distance. For example, the number of vertices that belong to the head is nearly the same as the number of vertices of limbs, while the latter has much more influence on the overall poses and thus should be placed larger weights during the distance calculation. Another issue is how to obtain the centers of each cluster after clustering. A direct average of vertices is obviously infeasible, since the resulting vertices are highly likely to not lie in the valid SMPL topology. To solve this issue, we choose to conduct on the samples’ SMPL parameters and apply separate averaging strategies on pose and shape parameters to suit their properties better.

In summary, we make the following contributions: We identify the relationship between monocular 3D human reconstruction models’ performances and samples’ distance to the applied singular human prototypes. To improve the models’ performance on tail classes, we design a prototype memory module to fully leverage the information contained across all samples of common and rare poses. Thanks to the unique formulation of local prototypes, our method improves the models’ performances on both tail classes and the overall samples. Our model achieves state-of-the-art performances on Human3.6M \[2\], MPI-INF-3DHP \[10\], 3DPW \[3\] and UP-3D \[11\]. In particular, the model’s average reconstruction errors on tail classes are reduced by 12 mm on challenging in-the-wild dataset, i.e. UP-3D \[11\]. The proposed method is easy to implement and can be adapted to various frameworks or other visual tasks that perform regression from mean parameters.

II. RELATED WORK

Single-Image 3D Human Reconstruction. Recent methods for single-image 3D human reconstruction share a similar pipeline. In particular, a parametric model SMPL \[2\] is used to represent 3D human, while DNN or optimization is employed to obtain the parameters or vertex coordinates of the SMPL model. Most of the learning-based methods \[5\], \[12\], \[13\], \[14\], \[15\], \[16\], \[17\] use CNN-based models to predict the parameters of SMPL directly. Kanazawa \textit{et al} \[5\] use adversarial losses \[18\] to judge the predicted 3D human poses. Pavlakos \textit{et al} \[12\] propose to predict the 2D poses heatmaps and silhouette as the intermediate representation to facilitate the model prediction. Güler \textit{et al} \[6\], Xu \textit{et al} \[13\] and Rong \textit{et al} \[14\] incorporate DensePose \[19\] into the framework. Moon \textit{et al} \[15\] design a framework to predict 3D poses first and regress the corresponding SMPL parameters based on the predicted 3D poses. Georgakis \textit{et al} \[16\] adopt a hierarchical strategy to predict the SMPL poses part by part, instead of regressing all together. Other learning-based methods adopt different frameworks. Kolotouros \textit{et al} \[7\] and Choi \textit{et al} \[20\] use graph convolution network \[21\] to predict SMPL vertices directly. Zeng \textit{et al} \[4\] regress location maps in the UV map space instead of the original SMPL parameter space.

Different from the aforementioned learning-based methods, optimization-based methods directly optimize the SMPL parameters instead of using DNNs. Bogo \textit{et al} \[22\] obtains SMPL parameters by minimizing the distances between ground-truth 2D keypoints and projected 2D keypoints from predicted SMPL models. Kolotouros \textit{et al} \[1\] designs a framework to unify the learning-based and optimization-based methods. They first deploy a CNN model to predict the SMPL parameters of the given images. The predicted SMPL parameters then serve as the initialization of optimization fitting \[22\]. After that, the optimized SMPL parameters are used to supervise the CNN model. Those steps are performed in iteration until convergence. Our model shares a similar framework as SPIN \[1\] while we use the fitted SMPL parameters to supervise the model instead of performing “in-the-loop” optimization in the training process. Another major difference is that we apply prototypical memory and assign each sample the closest prototype learned from data.

Using Prototypes in 3D Human Reconstruction. Most of the learning-based methods employ a 3D human prototype as the foundation to predict the required values. For example, several HMR \[5\]-based methods \[1\], \[13\], \[14\] use mean parameters calculated from datasets to represent the 3D human prototype. Güler \textit{et al} \[19\] use a set of Euler angles to form a convex hull as the basis for SMPL pose prediction. Kolotouros \textit{et al} \[7\] use mesh template as the input of a graph convolutional network \[21\] to directly regress 3D vertices of the SMPL model. DecoMR \[4\] also adopts the reference mesh as the base
for predicting location maps. Although these works achieve promising results on common poses, their performances drop drastically when applied to images with rare poses. It is because such methods only adopt one single 3D human prototype that is close to common poses while far from rare poses. Therefore, these methods are biased towards predicting common poses. To overcome this drawback, we propose to use multiple 3D human prototypes. Each data sample is assigned to the learned prototype that it is closest to. To our knowledge, LCR-Net [9] is the only work that applies multiple prototypes for 3D human estimation area. The prototypes used by LCR-Net is obtained from obtaining naive K-Means to 3D pose data. Nevertheless, obtaining effective prototypes for 3D human reconstruction is not that trivial. Directly applying naive K-Means on SMPL parameters, i.e., pose parameters and shape parameters, will lead to suboptimal performance. To circumvent this hurdle, we elaborately design a specified clustering algorithm for SMPL model, which is more effective in 3D human reconstruction.

### III. Methodology

To improve the models’ performances on tail classes, we propose a general framework called Prototypical Memory Network (PM-Net), as depicted in Fig. 2. The framework is composed of three parts: an encoder, a parameter estimator and a prototypical memory. The core component is a prototypical memory that stores multiple 3D human prototypes. Follow the practices of previous works [1], [5], we use mean SMPL parameters as the 3D human prototype. During inference, the encoder takes images as input and outputs feature vectors and prototype classification scores. The prototype classification scores are used to select mean parameters from the prototypical memory. After that, the feature vectors and selected mean parameters are fed into the parameter estimator. Lastly, the parameter estimator uses the selected mean parameters as the initialization and estimates the predicted parameters in an iterative fitting manner [1], [5].

In the following sections, we will first introduce the 3D human model used in this paper and the inference process of PM-Net. Next, we present the design of prototypical memory. Finally, we discuss the training strategy used in this work. The mathematical symbols used in this paper are listed in Table I.

#### TABLE I: List of mathematical symbols.

| Meaning       | Math Symbol |
|---------------|-------------|
| SMPL Pose Parameters | \( \theta \) |
| SMPL Shape Parameters | \( \beta \) |
| SMPL Parameters | \( \phi = (\theta, \beta) \) |
| SMPL Vertices | \( V \) |
| SMPL Joint Regressor | \( J \) |
| 3D Joints | \( J^{3D} \) |
| 2D Joints | \( J^{2D} \) |
| Keypoints Visibility | \( \mu \) |
| Cluster Center | \( \pi \) |
| Weights for each Vertex | \( W \) |
| Input Image | \( I \) |
| Encoder | \( E \) |
| Parameter Estimator | \( G \) |
| Prototypical Memory | \( M \) |
| Encoded Feature | \( f \) |
| Classification Score | \( c \) |
| Number of Clusters | \( K \) |

#### A. 3D Human Model

We use SMPL [8], a triangle-mesh-based model parameterized by the shape and pose parameters to represent a 3D human model. The shape parameters \( \beta \in \mathbb{R}^{10} \) influence the overall body shape. The pose parameters \( \theta \) model both the relative 3D rotations for a predefined kinematics skeleton with 23 joints and the global rotation for the whole body. The
original SMPL pose parameters are in angle-axis representation. Follow the practice of SPIN \cite{1}, we use the continuous rotation representations proposed by Zhou \textit{et al} \cite{23} to represent 3D rotations. Therefore, the dimension of pose parameters becomes $\theta \in \mathbb{R}^{6 \times 24}$. Given the shape and pose parameters, SMPL model calculates the coordinates of vertices $V \in \mathbb{R}^{3 \times 6890}$. Given SMPL vertices $V$, 3D joints $J^{3D} \in \mathbb{R}^{3 \times 24}$ are obtained by using predefined joint regressor $J \in \mathbb{R}^{3 \times 6890}$ by $J^{3D} = J \cdot V^{T}$.

B. Framework

The overall framework is shown in Fig. 2. A single image $I$ serves as the only input. The encoder $E$, composed of an image encoder, a feature encoder and a classifier, takes in input $I$ and outputs encoded feature $f$ and classification score $c$. The score $c$ is then used to selected matching mean parameters $\phi = (\tilde{\theta}, \tilde{\beta})$ from the memory $M$. Given the encoded feature $f$ and selected mean parameters $\phi$, the parameter estimator $G$ regresses the SMPL parameters $\phi$. In the last step, 3D meshes are generated by the SMPL model using the estimated parameters.

Contrary to the previous practices \cite{1, 5} that use the same mean parameters for all samples, our model assigns different sets of mean parameters to different samples. The framework maintains a prototypical memory that is composed of multiple sets of mean SMPL parameters. Given each sample, the matching set of parameters is selected. The number of mean parameter sets is fixed as $K$. These $K$ set of mean parameters further form the mean parameters memory $M \in \mathbb{R}^{K \times D}$, where $D$ is the dimension of the parameters. In data preprocessing, each data sample is assigned a one-hot class label $c \in \mathbb{R}^{1 \times K}$, indicating to which set of mean parameters the sample is closest to. During inference, the classifier in the encoder $E$ produces a classification score $c \in \mathbb{R}^{1 \times K}$ for each sample. With the classification score $c$, the mean parameters $\phi$ are selected from the prototype memory $M$ as $\hat{\phi} = cM$. The obtained $\hat{\phi}$ serves as the initialization for iteratively fitting the parameters as $\phi_{t+1} = \phi_t + \Delta \phi_t$, where $\phi_0 = \hat{\phi}$. Follow the practice of previous works \cite{1, 5}, fitting process is iterated for 3 times.

C. Building Prototypical Memory

\begin{algorithm}

\textbf{Algorithm 1 Part-Aware 3D Human K-Means}

\textbf{Require:} Threshold for average sample-to-center distance $\hat{\gamma}$; 
\textbf{Require:} Threshold for total number of iterations $\hat{\lambda}$. 
\textbf{Require:} N samples: $\Phi = \{\phi_1, \ldots, \phi_N\}$; $\phi_i \in \mathbb{R}^{154}$. 
\textbf{Require:} Initial clusters centers: $\Pi = \{\pi_1, \ldots, \pi_K\}$; $\pi_j \in \mathbb{R}^{154}$. 
\textbf{Require:} Weights for each vertex $W \in \mathbb{R}^{3 \times 6890}$. 

1: \textbf{procedure} CLUSTERING($\hat{\gamma}$, $\hat{\lambda}$, $\Phi$, $\Pi$, $W$) 
2: \hspace{10mm} Initialize sets of assigned samples $\Phi_i = \{\}$, $i = 1, \ldots, K$. 
3: \hspace{10mm} Initialize set of sample-to-center distances, $\Gamma = \{\}$. 
4: \hspace{10mm} Initialize average sample-to-center-distance $\hat{\gamma} = \infty$; 
5: \hspace{10mm} Initialize the number of iterations $\lambda = 0$.
6: \hspace{10mm} \textbf{while} $\hat{\gamma} < \hat{\gamma}$ or $\lambda < \hat{\lambda}$ \textbf{do}
7: \hspace{20mm} \textbf{for} $i \leftarrow 1, N$ \textbf{do} \hspace{10mm} $\triangleright$ Assign samples. 
8: \hspace{30mm} \textbf{for} $j \leftarrow 1, K$ \textbf{do} \hspace{10mm} $\triangleright$ Update cluster centers. 
9: \hspace{40mm} $V_i = \text{smpl}(\phi_i)$; $V_j = \text{smpl}(\pi_j)$; 
10: \hspace{40mm} $\gamma_{ij} = \| (V_i - V_j) \circ W \|_2$ 
11: \hspace{20mm} end for 
12: \hspace{20mm} $a_i = \arg \min_j (\gamma_{ij})$. 
13: \hspace{20mm} $a_i = \arg \min_j (\gamma_{ij})$. 
14: \hspace{20mm} Add sample $\phi_i$ to $a_i$-th cluster. 
15: \hspace{20mm} Add distance $\gamma_i$ to $\Gamma$. 
16: \hspace{20mm} end for 
17: \hspace{10mm} end for 
18: \hspace{10mm} \textbf{for} $j \leftarrow 1, K$ \textbf{do} \hspace{10mm} $\triangleright$ Update cluster centers. 
19: \hspace{20mm} $\Theta_j = \{\}$; $B_j = \{\}$. 
20: \hspace{20mm} \textbf{for} $\phi_{jk} \in \Phi_j$ \textbf{do} \hspace{10mm} $\triangleright$ Update cluster centers. 
21: \hspace{30mm} $(\theta_{jk}, \beta_{jk}) = \phi_{jk}$. 
22: \hspace{30mm} Add $\theta_{jk}$ to $\Theta_j$. 
23: \hspace{30mm} Add $\beta_{jk}$ to $B_j$. 
24: \hspace{20mm} end for 
25: \hspace{20mm} $\theta_{j} = \text{average}_\theta (\Theta_j)$. 
26: \hspace{20mm} $\beta_{j} = \text{average}_\beta (\Theta_j)$.
27: \hspace{20mm} $\pi_j = (\theta_{j}, \beta_{j})$. 
28: \hspace{20mm} end for 
29: \hspace{10mm} $\hat{\gamma} = \text{average}(\Gamma)$; \hspace{10mm} $\triangleright$ Update average distances. 
30: \hspace{10mm} $\lambda = \lambda + 1$. 
31: \hspace{10mm} end while 
32: \hspace{10mm} end procedure

\end{algorithm}

To construct the prototypical memory, we first apply clustering on training data. The obtained cluster centers are used to compose the memory. Instead of applying naive K-Means clustering algorithm on SMPL parameters $\phi = (\tilde{\theta}, \tilde{\beta})$, we elaborately design a clustering algorithm that exploits the characteristics of 3D human reconstruction. The algorithm is based on K-Means with following modifications:
Fig. 4: Visualization of 3D human prototypes in the prototypical memory. We visualize several representative sets of mean parameters stored in the prototypical memory. We render each selected mean parameters in three views: the original view, frontal view and side view. From the figure, it can be found that the prototypical memory covers both commons poses like standing ((a)) and rare poses such as squatting ((b)). It also contains both common view types (frontal view in (a) and back view in (c)) and rare view types, such as head upside down in (d).

1) Distance calculating – Instead of calculating the distance between parameters \( \phi \), we use the SMPL model to obtain the vertices \( V \) of the corresponding parameters \( \phi \). Then we calculate the Euclidean distance based on the vertex coordinates. We argue that this distance format is more suitable since it directly and effectively reflects the pose and shape variations across different samples. Furthermore, in calculating the distance based on vertices, we find it is suboptimal to assign the same weights to all the vertices. For example, the number of vertices belonging to the head is similar to the number of vertices of limbs, while the limbs have more influence on the overall pose estimation. Therefore, we assign larger weights to body parts that have more influences on pose estimation while smaller weights to the other parts. We call this strategy as part-aware weighting. In our experiment, the weights for vertices belonging to different body parts are empirically set as 5.0 for limbs (arms and legs), 0.3 for head and hand, 0.5 for foot, 1.0 for torso. Visualization of vertices belonging to different body parts is shown in Fig. 5.

2) Cluster center updating – To update centers of each cluster after each iteration, a straightforward idea is to directly average pose and shape parameters of the assigned samples. Averaging shape parameters is valid since the process of blending shapes is linear. Nevertheless, directly averaging pose parameters, which are in the form of 3D rotations, is not reasonable, as pointed out by previous works [24]. To obtain the valid averaged rotations, we first transfer the 3D rotations to the format of quaternions. Then we apply the quaternion averaging algorithm proposed by Markley et al [24].

To be more specific, suppose we have \( n \) quaternions \( q_i \) to be averaged, the averaged quaternions \( \bar{q} \) can be obtained by solve the following equations:

\[
M = \sum_{i=1}^{n} q_i q_i^T \\
\bar{q} = \arg\max_{q \in \mathbb{S}^3} q^T M q
\]

\( \mathbb{S}^3 \) is the 3 dimensional unit sphere. The solution of the maximization problem is the eigenvector of \( M \) which corresponding to the maximum eigenvalue. We refer the readers to the original paper [24] for more details.

Equipped with the aforementioned characteristics, we design a new clustering algorithm for 3D human reconstruction. We call it Part-Aware 3D Human K-Means (P3DH K-Means). The whole process of P3DH K-Means is composed of initialization, sample assignment and center updating. The number of clusters is fixed to be \( K \). Suppose we use \( N \) samples in performing the clustering. Each sample is represented as the concatenation of its pose parameters \( \theta \in \mathbb{R}^{6 \times 24} \) and shape parameters \( \beta \in \mathbb{R}^{10} \). The centers of each cluster \( \pi_j \) are represented in the same way. In the initialization step, we randomly select \( K \) samples from the training data and use them as the initial cluster centers. In cluster assignment step, for each sample, the distances of its SMPL vertices between the SMPL vertices of each cluster center are calculated. The sample is then assigned to the closest cluster based on the distances. In center updating step, shape parameters are calculated by directly averaging the shape parameters of the assigned samples. To calculate the pose parameters, the pose parameters of the assigned samples are first converted to quaternions format. Then the quaternion averaging algorithm proposed in [24] is used to obtain the averaged quaternions. After that, the averaged quaternions are
converted back to the continuous rotation representation [23]. The process of sample assignment and center updating iterates until the algorithm converges. The whole process of P3DH K-Means is presented in Algorithm 7. We further show several selected cluster centers in Figure 4. It is observed that the obtained cluster centers cover both common and rare poses with different view types.

D. Training Scheme

We employ 3D and 2D losses to train our model. The 3D loss is composed of SMPL pose parameter loss $L_{\theta}$, SMPL shape parameter loss $L_{\beta}$ and 3D joint loss $L_{J3D}$. These losses are defined as follows:

\[
L_{\theta} = \| \theta - \hat{\theta} \|_2^2, \\
L_{\beta} = \| \beta - \hat{\beta} \|_2^2, \\
L_{J3D} = \| J^{3D} - \hat{J}^{3D} \|_2^2, \\
\]

(2)

where $\hat{\theta}$, $\hat{\beta}$ and $J^{3D}$ are ground truth SMPL pose parameters, shape parameters and 3D Joints.

To estimate 2D keypoints, we adopt a weak perspective camera model that is composed of scale $s$ and camera translation $[t_x, t_y]$. The camera parameters $[s, t_x, t_y]$, are also predicted by the parameter estimator $G$. Given the camera parameters, 2D keypoints $J^{2D}$ are projected from $J^{3D}$. Then predicted $J^{2D}$ is used to calculate the 2D keypoint loss $L_{J2D}$ with the ground-truth 2D keypoints $\hat{J}^{2D}$. The whole process is formulated as below,

\[
J^{2D} = s \times \hat{J}^{3D} + t_x, \\
\hat{J}^{2D} = s \times \hat{J}^{3D} + t_y, \\
L_{J2D} = \| \mu \cdot (\hat{J}^{2D} - \hat{J}^{2D}) \|_2^2, \\
\]

(3)

where $\mu$ is the visibility indicator.

Since our model requires assigning samples into different prototypes, we apply a cross-entropy classification loss $L_C$, defined as:

\[
L_C = -\hat{c} \cdot \log(c). \\
\]

(4)

**Overall Loss Function.** The overall loss $L$ is defined as:

\[
L = \lambda_1 L_{J3D} + \lambda_2 L_{J2D} + \lambda_3 L_{\theta} + \lambda_4 L_{\beta} + \lambda_5 L_C. \\
\]

(5)

In experiments, we set $\lambda_1 = \lambda_2 = 5.0$, $\lambda_3 = 1.0$, $\lambda_4 = 1e^{-3}$ and $\lambda_5 = 1.0$. These hyper-parameters are obtained from grid search.

IV. EXPERIMENTS

In this section, we first introduce the experiment setup. Then we discuss the design of key features of the prototypical memory. After that, we compare our method with previous state-of-the-art methods. In the end, we provide further analysis on the design of PM-Net and typical failure cases.

### TABLE II: Ablation Study. Evaluating variants of prototypical memory. The evaluation metric is MPVPE.

| Dataset | Methods | Human3.6M | 3DPW | UP-3D |
|---------|---------|-----------|------|-------|
| SPIN    | 78.7    | 117.8     | 119.3|
| K-Means Naive | 79.1 | 116.6 | 115.1|
| Random Select | 76.9 | 115.9 | 114.1|
| 3DH K-Means | 75.4 | 115.8 | 113.5|
| P3DH K-Means | 73.6 | 114.6 | 112.6|

A. Experimental Setting

In our experiments, we employ several 3D human datasets: Human3.6M [2], MPI-INF-3DHP [10], 3DPW [3], UP-3D [11]. To increase the generalization ability of the models, we also adopt several in-the-wild 2D datasets, including LSP-Extended [25], MPII [26] and COCO [27]. In our experiments, we use Human3.6M, MPI-INF-3DHP, MPII, LSP-Extended and COCO for training. Our models are evaluated on Human3.6M, MPI-INF-3DHP, 3DPW and UP-3D. In the following of this subsection, we first discuss the evaluation metrics used in our work. Then we give a brief introduction to each dataset. In the end, we provide the implementation details.

**Evaluation Metrics.** In this paper, we mainly use mean per-vertex position error (MPVPE) [13], [14] as the evaluation metric. We believe MPVPE is more suitable than joint-based errors such as MPJPE, since they only consider joint positions while neglecting the joint rotations and body shapes. For completeness, we also adopt mean per-joint position error (MPJPE) and PA-MPJPE, which is MPJPE after applied Procrustes Analysis [28]. These two metrics are used to compare with the previous works and evaluate on datasets without ground-truth SMPL parameters, such as MPI-INF-3DHP [10].

**Human3.6M.** Human3.6M [2] is an indoor dataset with 3D joint annotations. We use Mosh [29] to collect SMPL parameters from 3D Mocap markers. We train our models on subject S1, S5, S6, S7, and S8 and evaluate them on S9 and S11. Following the practice of previous works [5], [1], we only evaluate on samples of frontal camera (camera 3). This evaluation strategy is often called “Protocol-2” in the literature.

**MPI-INF-3DHP.** The dataset [10] is captured with a multi-view camera system under a controlled environment. We use the subject S1 to S8 for training and evaluate our models on the evaluation set. The original dataset only provides 3D joints. In training, we also use the pseudo ground truth SMPL parameters provided by SPIN [1] through multi-view fitting.

**3DPW.** Data of 3DPW [3] is captured under in-the-wild scenarios. The ground truth SMPL pose parameters are obtained from IMUs while shape parameters are obtained from the 3D scan. To enable fair comparisons with previous methods, we only use 3DPW’s test set for evaluation and do not train our models on it.

**UP-3D.** UP-3D [11] is composed of selected images from four 2D pose datasets. An extended SMPLify [22] is used to predict SMPL parameter on those images and then human annotators are asked to select the good samples. We only use UP-3D’s test set for evaluation and do not train our model on it.
Fig. 5: Ablation study. We show the performances of four different variants of the proposed PM-Net, namely K-Means Naive, Random Center, 3DH K-Means, and P3DH K-Means. We evaluate on the evaluation set of Human3.6M [2], 3DPW [3] and UP-3D [11]. Each evaluation set is divided into non-overlapping testing sets according to the samples’ distances to the singular human prototype visualized in Fig. 1 (a). The x-axis of each sub-figure is the samples’ distances to the singular prototype. The y-axis is the models’ decreased prediction errors compared to the baseline model (SPIN [1]).

2D Pose Datasets. To increase the generalization ability of the trained models, we additionally train our models on several in-the-wild datasets, including LSP-Extended [25], MPII [26] and COCO [27]. These datasets only provide 2D landmark annotations. We use the SMPL fitting results provided by SPIN [1].

Implementation Details. All images are cropped around humans and scaled to 224×224. The image encoder is composed of ResNet-50 [30] with two additional fully connected layers, of which the output dimensions are both 1024. The classifier and parameter estimator are both fully connected layers, whose output dimensions are $K$ and 157, respectively. $K$ equals the number of 3D human prototypes in the prototypical memory, which is fixed to be 50 throughout our experiments unless specified. The 157 dimensions are composed of shape parameters (10), pose parameters (6×24) and camera parameters (3). The batch size is 256. Adam optimizer [31] with a learning rate $1e^{-4}$ is employed to train the models.

B. Design of Prototypical Memory

We use SPIN [5], which only adopts a single prototype, to serve as the baseline. Apart from reporting results on the overall evaluation set, we also split the evaluation set
**Fig. 6:** Qualitative comparison between PM-Net and baseline model. We qualitatively compare the PM-Net with previous state-of-the-art methods, SPIN [1]. The images are selected from Human3.6M [2] (Row 1), 3DPW [3] (Row 2) and UP-3D [11] (Row 3).

**TABLE III: Comparison with state-of-the-art methods (whole evaluation sets).** We compare proposed PM-Net with previous state-of-the-art methods on several widely used dataset including Human3.6M [2], 3DPW [3], UP-3D [11] and MPI-INF-3DHP [10]. We use MPVPE, MPJPE and PA-MPJPE as the evaluation metrics. We mark the methods that use UP-3D in training with *. It is worth noting that our models are not trained on UP-3D.

| Dataset & Metrics → Methods ↓ | Human3.6M [2] | 3DPW [3] | UP-3D [11] | MPI-INF-3DHP [10] |
|---|---|---|---|---|
| | MPVPE | MPJPE | PA-MPJPE | MPVPE | MPJPE | PA-MPJPE | MPVPE | MPJPE | PA-MPJPE |
| Pavlakos et al [12] | - | 75.9 | - | 117.7 | - | - |
| CMR* [4] | 87.1 | 71.9 | 50.1 | 144.2 | 127.6 | 74.6 | 96.2 | - | - |
| DecoMR* [4] | 78.6 | 59.6 | 39.1 | 141.4 | 126.6 | 73.7 | 93.4 | - | - |
| HMR [15] | - | - | 56.8 | - | 81.3 | - | 124.2 | 89.8 |
| DenseRac [13] | - | - | 48.0 | - | - | - | 114.2 | 83.5 |
| SPIN [1] | 78.8 | 62.5 | 41.1 | 117.8 | 98.3 | 60.2 | 119.3 | 105.2 | 67.5 |
| HKMR [16] | - | 60.0 | - | - | - | - | - | - |
| I2L-MeshNet [15] | - | 55.7 | 41.7 | - | 95.4 | 58.6 | - | - |
| Pose2Mesh [20] | - | 67.9 | 49.9 | - | 91.4 | 60.1 | - | - |
| Ours | 73.6 | 59.6 | 40.6 | 114.6 | 94.6 | 58.4 | 112.6 | 98.2 | 62.8 |

**TABLE IV: Comparison with state-of-the-art methods (tail classes).** We compare with several state-of-the-art methods on tail classes on Human3.6M [2], 3DPW [3] and UP-3D [11]. “Tail-𝑥%” means to only evaluate the data samples with the largest 𝑥% distances to the singular prototype as depicted in Fig. 1 (a). The evaluation metric is MPVPE.

| Dataset & Metrics → Methods ↓ | Human3.6M [2] | 3DPW [3] | UP-3D [11] |
|---|---|---|---|
| | Tail-5% | Tail-10% | Tail-5% | Tail-10% | Tail-10% | Tail-20% |
| CMR* [4] | 92.5 | 97.3 | 209.9 | 194.3 | 170.4 | 149.2 |
| DecoMR* [4] | 84.9 | 88.2 | 184.3 | 175.1 | 162.0 | 142.5 |
| SPIN [1] | 89.0 | 88.8 | 130.0 | 130.6 | 184.2 | 167.9 |
| Ours | 80.0 | 81.1 | 124.9 | 126.4 | 172.1 | 158.6 |

of Human3.6M [2], 3DPW [3] and UP-3D [11] into un-overlapping subsets, according to samples’ distances to the singular prototype, as depicted in Fig. 1 (a). We then report models’ reconstruction errors (MPVPE) on each subset.

In our experiments, we evaluate four types of prototypical memory, including K-Means Naive, Random Center, 3DH K-Means and P3DH K-Means. “K-Means Naive” means directly applying the original K-Means algorithm on the SMPL parameters (pose parameters and shape parameters). “Random Center” uses initial centers, which are randomly selected from the samples, as the final cluster centers. The samples are assigned to clusters using the vertex-based distance without applying part-aware weighting. “3DH K-Means” refers to the P3DH K-Means without applying part-aware weighting. These four different models share the same pipeline depicted in Fig. 2. The training process and other hyperparameters, such as the number of clusters, are also the same. The only differences are how the prototypical memories are built and how samples are assigned to the corresponding prototypes. The evaluation results are shown in Fig. 5. The x-axis of each sub-figure is the samples’ distances to the singular prototype depicted in Fig. 1 (a). The y-axis is the models’ error reduction compared to the baseline model (SPIN [1]). The error is MPVPE. Besides, we also list evaluation results on the whole evaluation set Table III.

It is revealed that P3DH K-Means achieves the best performance on nearly all the subsets of all the three datasets, especially on tail classes that are far from the singular human prototype. 3DH K-Means performs better than Random Center,
proving that performing clustering to get better sample assignment is beneficial. Furthermore, it is worth noting that directly performing naive K-Means clustering on SMPL parameters is not effective. It is even not comparable to Random Center. The experimental results prove the significance of performing the 3D human-specific clustering.

We further show qualitative results in Fig. 6 to compare with the baseline model (SPIN [1]). The results show that our model can generate more precise results on challenging samples with rare poses such as squatting on the ground or uncommon views such as head upside down. Results listed in Fig. 4 are generated by PM-Net using P3DH K-Means to build the prototypical memory.

C. Main Results

Comparison with state-of-the-art methods. In this subsection, we compare our model with previous state-of-the-art methods. For methods with code released, we use their original code and models to obtain the evaluation results that are not included in their original papers. For other methods, we report the original results listed in their papers. We report evaluation results on both the whole evaluation sets and tail classes only. The results are listed in Table III for the overall evaluation results and Table IV for tail class evaluation, separately. In Table IV “Tail-x%” means to only evaluate samples with the largest x% distances to the singular prototype as depicted in Fig. 1 (a). The results in Table III show that our model achieves comparable performances with previous state-of-the-art methods on the whole evaluation sets. Furthermore, results in Table IV show that our model achieves better results on tail classes. These experimental results show that the proposed PM-Net effectively improves the models’ performances on challenging tail classes while achieving comparable or better performance on the samples.

Qualitative Results. We show several qualitative results in Fig. 8. It is demonstrated that the obtained prototypical memories contain both common poses and rare poses. Furthermore, the proposed PM-Net can effectively select the matched prototypes for the input images and use them as the foundation to generate precise predictions.

D. Further Analysis

In this subsection, we evaluate two designs of PM-Net that influence the models’ performance. 1) The number of prototypes in prototypical memory. 2) The weight of limbs in performing part-aware weighting. The experiments are conducted on the evaluation set of Human3.6M [2] with the same evaluation protocol defined in section IV-C. The evaluation metric is MVPVE. All the models reported in this subsection adopt P3DH K-Means to build the prototypical memory.

Influence of the Number of Prototypes. We first study the influence of the number of prototypes (denoted as K). In our experiments, K ranges from 5 to 1000. The results are listed in Table V. The highest performance is achieved when K equals to 50. When K is smaller than the optimal value, the capacity of prototype memory is not sufficient to provide the best-suited prototype for each data sample. When K is larger than the optimal values, the data samples assigned to each prototype will decrease and becomes insufficient to train the conditional regressor.

Influence of the Limb Weights. We next study the influence of the limb weights in performing part-aware clustering. We also do experiments to examine the influence of weights of other body parts. It turns out the models’ performances only differ slightly. Therefore, we only focus on the influence of limb weights in this subsection. The weights of other body part is set to be value shown in Fig. 5. In our experiments, the weight of limb ranges from 1.0 to 10.0. The results are listed
in Table VI. The highest performance is achieved when the weight is set to be 5.0. When the weight of limbs is smaller than the optimal value, the vertices of limbs are not assigned enough importance. When the weight of limbs is larger than the optimal values, too much attention is paid to the limb, therefore affecting the learning of other parts.

V. CONCLUSION

In this paper, we propose Prototypical Memory Network (PMNet) to mitigate current single-image 3D human reconstruction models’ suboptimal performance on rare poses. The key of the proposed method is a prototypical memory that learns and stores a set of 3D human prototypes that can capture compact local distributions to represent both common and rare poses. It lifts the burden from the parameter regressor in coping with diverse poses. While we mainly focus on the task of 3D human reconstruction in this paper, the notion of Prototypical Memory can be extended to other visual tasks that need to handle data with multiple modes.

REFERENCES

[1] N. Kolotouros, G. Pavlakos, M. J. Black, and K. Daniilidis, “Learning to reconstruct 3D human pose and shape via model-fitting in the loop,” in IEEE International Conference on Computer Vision, 2019.

[2] C. Ionescu, D. Papava, V. Olaru, and C. Sminchisescu, “Human3.6M: Large scale datasets and predictive methods for 3D human sensing in natural environments,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 36, no. 7, pp. 1325–1339, 2014.

[3] T. von Marcard, R. Henschel, M. Black, B. Rosenhahn, and G. Pons-Moll, “Recovering accurate 3D human pose in the wild using IMUs and a moving camera,” in European Conference on Computer Vision, 2018.

[4] W. Zeng, W. Ouyang, P. Luo, W. Liu, and X. Wang, “3D human mesh regression with dense correspondence,” in IEEE Conference on Computer Vision and Pattern Recognition, 2020, pp. 7054–7063.

[5] A. Kanazawa, M. J. Black, D. W. Jacobs, and J. Malik, “End-to-end recovery of human shape and pose,” in IEEE Conference on Computer Vision and Pattern Recognition, 2018.

[6] R. A. Guler and I. Kokkinos, “Holopose: Holistic 3D human reconstruction in-the-wild,” in IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 10 884–10 894.

[7] N. Kolotouros, G. Pavlakos, and K. Daniilidis, “Convolutional mesh regression for single-image human shape reconstruction,” in IEEE Conference on Computer Vision and Pattern Recognition, 2019.

[8] M. Loper, N. Mahmood, J. Romero, G. Pons-Moll, and M. J. Black, “SMPL: A skinned multi-person linear model,” Transcation on Graphics, 2015.

[9] G. Rogez, P. Weinzaepfle, and C. Schmid, “Lcr-net++: Multi-person 2D and 3D pose detection in natural images,” tpami, vol. 42, no. 5, pp. 1146–1161, 2019.

[10] A. Mehta, H. Rhodin, D. Casas, P. Fua, O. Sotnychenko, W. Xu, and C. Theobalt, “Monocular 3D human pose estimation in the wild using improved CNN supervision,” in International Conference on 3D Vision. IEEE, 2017, pp. 506–516.

[11] G. Lassner, J. Romero, M. Kiefel, F. Bogo, M. J. Black, and P. V. Gehler, “Unite the people: Closing the loop between 3D and 2D human representations,” in IEEE Conference on Computer Vision and Pattern Recognition, 2017.

[12] G. Pavlakos, L. Zhu, X. Zhou, and K. Daniilidis, “Learning to estimate 3D human pose and shape from a single color image,” in IEEE Conference on Computer Vision and Pattern Recognition, 2018.

[13] Y. Xu, S.-C. Zhu, and T. Tung, “Densercal: Joint 3D pose and shape estimation by dense render-and-compare,” in IEEE International Conference on Computer Vision, 2019.

[14] Y. Rong, Z. Liu, C. Li, K. Cao, and C. C. Loy, “Delving deep into hybrid annotations for 3D human recovery in the wild,” in IEEE International Conference on Computer Vision, 2019.

[15] G. Moon and K. M. Lee, “12l-meshnet: Image-to-lexel prediction network for accurate 3D human pose and mesh estimation from a single RGB image,” in European Conference on Computer Vision, 2020.
Fig. 8: **Qualitative Results.** Qualitative results of PM-Net and P3DH K-Means. The images are selected from Human3.6M [2], 3DPW [3] and UP-3D [11]. For each selected sample, we show the selected prototype, input images and PM-Net’s prediction. Each prototype is rendered from original view, frontal view and side view.