GaitSTR: Gait Recognition with Sequential Two-stream Refinement

Wanrong Zheng*, Haidong Zhu*, Zhaoheng Zheng, and Ram Nevatia, *Fellow, IEEE*

Abstract—Gait recognition aims to identify a person based on their walking sequences, serving as a useful biometric modality as it can be observed from long distances without requiring cooperation from the subject. In representing a person's walking sequence, silhouettes and skeletons are the two primary modalities used. Silhouette sequences lack detailed part information when overlapping occurs between different body segments and are affected by carried objects and clothing. Skeletons, comprising joints and bones connecting the joints, provide more accurate part information for different segments; however, they are sensitive to occlusions and low-quality images, causing inconsistencies in frame-wise results within a sequence. In this paper, we explore the use of a two-stream representation of skeletons for gait recognition, alongside silhouettes. By fusing the combined data of silhouettes and skeletons, we refine the two-stream skeletons, joints, and bones through self-correction in graph convolution, along with cross-modal correction with temporal consistency from silhouettes. We demonstrate that with refined skeletons, the performance of the gait recognition model can achieve further improvement on public gait recognition datasets compared with state-of-the-art methods without extra annotations.

Index Terms—Gait Recognition, Skeletons Representation, Joint and Bone, Temporal Refinement.

1 Introduction

Gait recognition [1], [2], [3], [4] is to identify the person present in a walking sequence. Different from other modalities, gait has the advantage of being able to be observed from a long distance and without the subject's cooperation. For gait recognition, researchers have developed silhouette-based methods, such as GaitSet [5], GaitPart [6], GaitGL [7], *etc.*, and skeleton-based methods like GaitGraph [8]. However, both input modalities exhibit certain deficiencies. Binarized silhouettes suffer from variations due to clothing and carried objects, as shown in Figure 1 (a), introducing external ambiguity, with segmented parts of a binarized silhouette being unavailable. Skeletons, on the other hand, include inconsistencies across frames in a sequence due to erroneous joint predictions, as depicted in Figure 1 (b), thereby reducing the accuracy of gait recognition.

In this paper, we propose the fusion of silhouette sequences with skeletons, harnessing the advantages of both modalities by refining the skeletons using silhouette sequences. Given that jitters in the detected skeletons are confined to a few frames isolated from the entire sequence, they lack temporal consistency with their neighboring frames [9]. Simple temporal smoothing, however, can introduce further confusion for gait recognition as the generated skeletons create new poses inconsistent with the current sequence. On the other hand, silhouettes for neighboring frames exhibit better temporal consistency due to minor changes in adjacent image conditions. We enhance the quality of skeletons by employing silhouettes to rectify the jitters while retaining necessary identity information for more accurate gait recognition.

We introduce *GaitSTR*, a **S**equential **T**wo-stream **R**efinement method, to refine the skeletons and combine them with silhouettes for gait recognition based on GaitMix and GaitRef [10]. When extracting the silhouette and skeletons from the same walking sequence, the temporal consistency between the two modalities is capable of providing guidance for each other: when silhouettes are not preserving useful boundaries of the images, skeletons can furnish the positions of the joints for pose estimation and recognition, and when the detection results of skeletons are unreliable, silhouettes can provide the pose information for the current frame in the sequence.

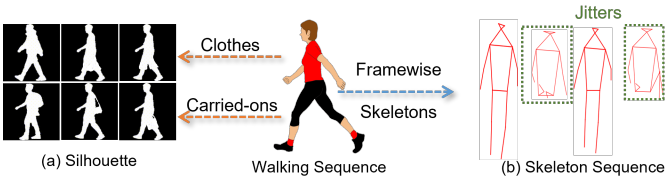

Fig. 1. Visualization of the (a) silhouette and (b) skeleton sequence used for gait recognition. Silhouettes show different contours with different clothes and carried-on objects, while the skeletons suffer from jittery detection results in the video.

To refine the skeletons, we introduce two-level fusion: an internal fusion within skeletons and a cross-modal correction with the temporal guidance from the silhouettes describing the same walking sequences. As skeletons can be decoupled into two different representations [11], [12], joints, and bones, we incorporate self-correction between the frame-wise joints and bones for increased consistency. Introduction of the bones, in addition to joints, is to provide more connectivity as the GCN [13] primarily focuses on the position of the joints and does not explicitly explore the distances between the nodes other than binarized connectivity. To refine these two representations, we incorporate two different spatial-temporal graph convolution branches [13] for these two modalities, with the same number of layers and dimensions at each level. After each graph convolution operation, we utilize a

- *The first two authors contributed equally.*
- *W. Zheng, H. Zhu, Z. Zheng and R. Nevatia are with the Department of Computer Science, University of Southern California, CA, 90089. E-mail: {wanrongz,haidongz,zhaoheng.zheng,nevatia}@usc.edu.*

Manuscript received November 14, 2023.

self-correction residual block to forward the information of the joints and bones, and add the information to the other branch residually [14] between the same level of layers.

In addition to the internal fusion of skeletons, we further introduce the cross-modal fusion between the silhouette and skeletons, combining the encoded silhouette features with the encoded frame-wise skeleton features to predict the relative changes for joints and bones in the skeletons. Since the gait pattern should be consistent for the same person, features from the silhouettes and skeletons describing the same walking sequence should also be consistent, facilitating the refinement of the skeletons with encoded silhouette features. Moreover, the sequence-level silhouette feature aids the frame-level skeletons for each frame in understanding its corresponding poses without losing identity information, as the temporal feature for the person is consistent and shared across all the frames in the same walking sequence.

With the predicted changes to the points, we reintegrate them into the original skeleton sequence and employ the skeleton encoder to extract the skeleton feature. We then concatenate this feature with the silhouette feature to predict the identity of the sequence using the refined skeletons. We compare *GaitSTR* with baseline multimodal gait recognition methods that use skeletons and silhouettes, including GaitMix [10], which concatenates the silhouette and skeleton features, and GaitRef [10], which uses silhouette features to refine joints. We show that skeleton refinement across the skeleton and silhouettes aids the final gait recognition, while adding internal correction within skeletons yields the best performance. We assess our method on four public datasets: CASIA-B [15], OU-MVLP [16], Gait3D [17], and GREW [18]. Our findings demonstrate that the refined skeletons, when combined with silhouettes, outperform other state-of-the-art gait recognition methods that utilize skeletons and silhouettes.

In summary, our contributions are: 1) we introduce *GaitSTR* which combines skeletons and silhouettes in an end-to-end training framework for gait recognition networks, 2) we incorporate the two different representations, joints and bones, for enhanced skeleton correction through self-fusion within skeletons, and 3) we use the consistency between silhouettes and skeletons to assist in correcting jitters in skeletons without additional supervision.

This paper is an extension of a conference version paper [10]. The novel contributions of this work are as follows.

1) In addition to using joints as the skeleton representations in GaitMix and GaitRef [10], we jointly utilize joints and bones as the representations. Unlike the single modal, the joints and bones represent different attributes of the skeletons, complementing each other.
2) Besides the cross-modal fusion between silhouette and skeletons for refinement, we also introduce fusion between joints and skeletons. We demonstrate that feature integration and refinement provide a more comprehensive understanding between each level of features, and yield more consistent feedback to the skeleton representation, which results in improved gait recognition accuracy.
3) We include additional experiments show that *GaitSTR*, an extension of GaitRef for the contribution of different modalities in skeletons and silhouettes.

2 Related Work

Gait Recognition aims to ascertain the corresponding identity of a person from their walking pattern. Given privacy concerns associated with RGB images, gaits are typically captured as two representations, silhouettes [15], [16], [19], [20], [21], [22], [23] and skeletons [10], [24], [25], [26], [27], [28]. Silhouettes document the boundary map of human segmentation. To mitigate the impact of appearance variants on human shapes, researchers have focused on part-based and body-shape reconstruction methods for gait recognition. GaitSet [5] and GLN [29] introduce set pooling and extract set features in the sequence. GaitPart [6] and GaitGL [7] partition the image into various small patches, utilizing local features to reduce the impact of appearance variants. Beyond directly mining identity information from silhouettes, ModelGait [30], Gait3D [17], PSE [31] and Gait-HBS [32] focus on 3-D shape reconstruction to assist identification from sequences. DyGait [20] and GaitGCI [19] explore temporal information between different frames to better model and understand dynamic gait patterns.

In addition to mining identity from silhouette sequences, some researchers [8], [33] have focused on using skeletons instead of silhouettes for gait recognition. Compared to the body contours represented by silhouettes, skeletons only encompass the joints and can eliminate the impact of body shapes as well as the appearance of the person. GaitGraph [8] employs the HRNet [34] for joint detections and utilizes the generated pose sequence for recognition. PoseGait [33] segregates the gait sequence into pose, limb, angle, and motion, subsequently analyzing the movements for each skeleton based on these four features independently before amalgamating them for gait recognition. In combining silhouettes and skeletons, Wang *et al.* [35] directly concatenate the two features, which still endures erroneous joint detections. Some existing works, such as GaitRef [10] and GaitMixer [36], discuss the combination and integration of silhouettes with skeletons directly. In our work, we explore different levels of temporal fusion within skeletons and across various modalities.

Pose Estimation and Refinement focus on extracting the human body poses and refinement. With the development of transformers, pose estimation is also transforming from CNN-based networks [37], [38] to transformer backbone networks [39], [40], [41], [42]. Pose estimation has experienced rapid development from CNNs [37] to vision transformer networks. Early works treat the transformer as a better decoder [39], [40], [41], [43]. Although the frame-level pose estimation accuracy is becoming more and more accurate, directly applying these methods to tasks with solid temporal relations, such as gait recognition, may introduce extra uncertainty with inaccurate joint predictions. For the sequence with strong temporal patterns, HuMoR [44] corrects the joint prediction of the person with the previous pose, and SmoothNet [9] filters the jitters in the whole sequence with analysis for the first and second deviation of the position for each point. These methods can fix some slight jitters and noise in the long sequence but still suffer when the poses for a long sequence are inaccurate. For the task of gait recognition with temporal repeated patterns, even with inaccurate predictions for the long sequence, the model should still fix the joints with the consistent moving pattern of the same person, which these existing methods cannot achieve.

3 Method

To recognize the person's identity, we combine silhouettes and skeletons for recognition. For silhouettes, we use the binarized body boundary images as input, and for skeletons, we take both bones and joints into consideration. Motivated by the bones used in skeleton-based action recognition [12], [45], the introduction of

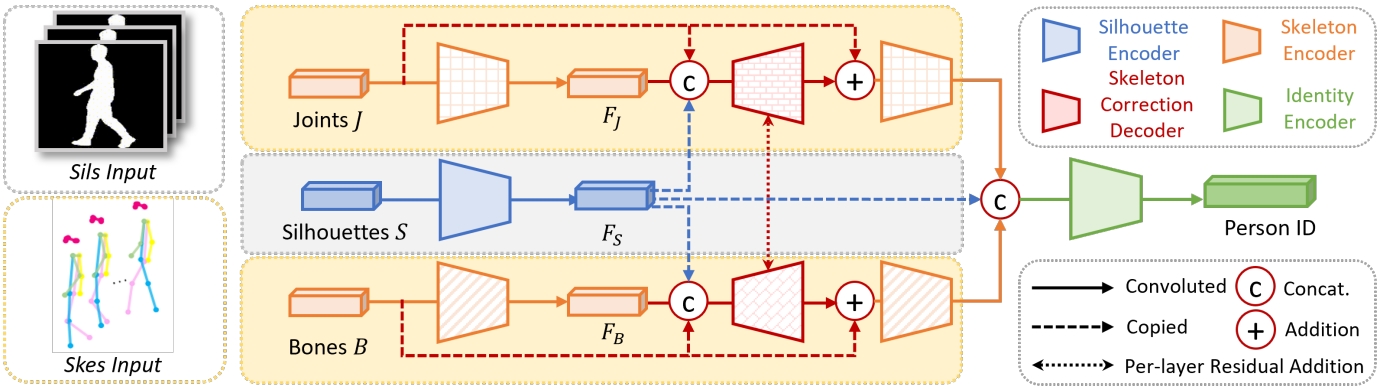

Fig. 2. Our proposed architecture for GaitSTR. Trapezoids consists of trainable modules, and modules of the same color and fill-in patterns in the same model share the weights. Dashed lines represent the operation of feature copying. S, J, and B are the input silhouettes, joints, and bones, respectively. F_S represents silhouette features, while F_J and F_B represent joint and bone features for skeleton representations.

bones emphasizes on the connections between the joints, while the joint-based graph convolution focuses largely on the nodes instead of the connection between them.

Given silhouettes S along with the joints J and bones B of the skeleton for the person p, gait recognition is to match the identity with the people in a pool $P = \{p_n\}_{n=1,2,...}$, where n is the candidate identity. We encode S, J and B to their corresponding embeddings and concatenate them together to find the nearest sample in P in the embedding space. In this section, we first discuss the baseline combination of the bones in addition to the joints and silhouettes in Section 3.1. We then present our proposed method *GaitSTR* to refine the input skeleton for gait recognition in Section 3.2, along with objectives and details for training in Section 3.3. We show our proposed architecture in Figure 2.

3.1 Baseline: Multimodal Gait Recognition

We build the multimodal gait recognition model as our baseline to combine information from different input modalities, including skeleton, joint and silhouettes. We employ two encoders: a silhouette feature encoder designed for encoding the silhouette S, and a skeleton feature encoder tasked with projecting the dual representations of the input raw skeletons, namely joints J and bones B, into their corresponding embedding spaces. Features generated from these two encoders are concatenated together and used for gait recognition.

Silhouette Feature Encoder. To extract the identity features from input sequential silhouette sequences, we use a silhouette feature encoder to convert the input silhouette sequence S to the corresponding output identity feature F_S. We have three steps for the silhouette feature encoder: convolution feature extraction, temporal pooling, and horizontal pooling. With the binary silhouette input sequence $S = \{s_i\}_{i=1,..,t}$, where i is the temporal stamp and t is the overall frame number, we apply a convolution network to extract the framewise feature f_i at frame i. f_i is an M-by-N-by-C matrix, where M and N are the height and width of the convoluted output features, and C is the channel number from the output of the last convolution layer.

With the framewise feature f_i, we use a max pooling layer for the temporal fusion and combine the feature into a single M-by-N-by-C output as temporal pooling. Since f_i still includes the spatial features for each segment, we follow [46] and apply horizontal pyramid pooling with scale P as 5. The output of the feature is a 2^{P-1}-by-C feature vector after horizontal pooling. The architecture of each component can be found in the implementation details in Section 4.1.

Skeleton Feature Encoder. In addition to the silhouette encoder, we deploy a skeleton feature encoder in parallel. This encoder processes the input skeleton sequences, consisting of joints $J = \{j_i\}_{i=1,...,t}$ that record the position of each keypoint of the body, and bones $B = \{b_i\}_{i=1,...,t}$ that are represented as vectors denoting the numerical directional relationship between two connecting joints, into their associated embeddings denoted as F_J and F_B. To represent both the joint and bone branches, we utilize two identical graph convolution networks; however, these networks do not share weights.

With the N-by-K-by-2 matrix to depict the 2-D skeletons of each frame, where K (either K_J for joints or K_B for bones) represents the number of points or connections of the skeletons, we implement a multi-layer spatial-temporal graph convolution network [13] for graphical feature extraction for each of them. By transforming the input from dimensions N-by-K-by-2 to N-by-K-by-C as the pre-frame per-node feature matrix, we then conduct average pooling over both the temporal and node dimensions and produce two final 1-by-C-length vectors, F_J and F_B, representing the features of the sequential skeleton pertaining to the input joints and bones. We concatenate these two vectors along their first dimension and generate a 2-by-C feature as the output of the skeleton feature encoder, along with the 2^{P-1}-by-C silhouette feature to represent the input sequence for recognition.

3.2 GaitSTR: Sequential Two-stream Refinement

In addition to fusing features from skeletons and silhouettes for gait recognition, *GaitSTR* introduces encoded features from silhouettes to improve the temporal consistency of skeletons, thereby enhancing the quality of skeletal data. The consistency across the two representations addresses framewise jitters in skeleton generation, ensuring a smoother and more accurate skeletal representation. Conversely, refined skeletons contribute to the silhouette analysis by minimizing the impact of appearance variants on gait recognition. Besides the two primary encoders employed for multimodal gait recognition, *GaitSTR* incorporates two additional modules: a skeleton correction network, which rectifies framewise

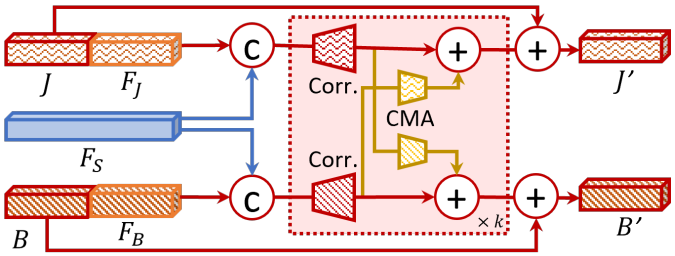

Fig. 3. Architecture of the skeleton correction network. F_J and F_B represent the joint and bone frame-wise features encoded from J (joints) and B (bones), respectively. The symbol 'C' denotes concatenation, and the plus sign denotes addition. 'Corr' refers to the skeleton correction network, while 'CMA' stands for the layer-level cross-modal adapter, and k denotes the number of layers over which cross-modal skeleton correction operations are repeated between bones and joints.

errors in skeleton predictions for joints and bones, and a cross-modal adapter, which bridges the gap between joint and bone representations to bolster the robustness of gait recognition.

Skeleton Correction Network. With information from the joint and bone feature, we use three different features as the network's input to correct the skeleton and compute the corresponding adjustment for each point: 2^{P-1}-by-C silhouette features F_S, N-by-K-by-C skeleton feature before pooling, and the original N-by-K-by-2 joint or bone matrix J or B. F_S provides the sequential information to correct the joint features F_J and F_B. F_J and F_B provide the framewise and feature for each node to correct the corresponding position of the joint in the frame. J and B provide the input order of the points to ensure the input and output order of the points are the same.

We show the architecture of the skeleton correction network in Figure 3. With these three inputs, we first flatten the silhouette feature into a $2^{P-1} \times C$ vector. We then repeat it N-by-K times and concatenate it with the other two features to form a N-by-K-by-$(2^{P-1} \times C + C + 2)$ feature matrix. To decode the new position J' for each node in the sequence, we decode the ΔJ for all the points with a reversed spatial-temporal graph convolution network to decode the N-by-K-by-2 adjustment for each node in J, and we have J' for refining the individual points in J following

$$J' = J + \Delta J = J + SkeletonDecoder(J, F_S, F_J). \quad (1)$$

The use of addition instead of directly predicting the corresponding location of the refined joints can give a relatively easier task for refinement and can preserve most of the original locations [47], since the original position of the joint has most of the sequential information correct and complete. By adding ΔJ on J, we get the final refined nodes as output and process it for further encoding.

Likewise, in the bone stream, we employ the same correction network to produce the adjustments, denoted as ΔB, on the original bone matrix B. This refinement follows a similar procedure as in the joint stream, formulated as:

$$B' = B + \Delta B = B + SkeletonDecoder(B, F_S, F_B). \quad (2)$$

With this refinement, we obtain the refined bone matrix B' with the assistance of encoded silhouette features, which is then utilized for further encoding in the network.

Cross-Modal Adapter. As the bones and joints represent the same skeleton and have connections between them, refining the skeleton and joints are also relevant to each other. We introduce the cross-modal adapter, $\text{CMA}_{i \to j}$, between these two modalities, as illustrated in Figure 3, where i and j represent the source and target modalities. As the correction network includes multiple layers as input and their architectures are similar, for each decoded skeleton representation F_B^x and F_J^x at layer x, we employ a two-layer MLP to project the features for the other modality following

$$\begin{aligned} F_B^x &= F_B^x + CMA_{J \to B}(F_J^x) \\ F_J^x &= F_J^x + CMA_{B \to J}(F_B^x). \end{aligned} \quad (3)$$

The refined feature F_B^x will be used for the input of the next graph convolution layer $(x+1)$ to enhance inter-modal communication for more robust and accurate gait recognition.

After we get the refined skeleton J' and B', we apply the same skeleton feature encoder in Section 3.1 and apply it on the refined skeleton sequence for predicting the 1-by-C skeleton feature $F_{J'}$ and $F_{B'}$. The two skeleton feature encoders for each modality share the parameters to ensure the two embedding spaces are the same between F_J and $F_{J'}$ as well as between F_B and $F_{B'}$. Using the same skeleton feature encoder can also extend the data available for the encoder training to train a stabler graph convolution model the skeleton feature extraction.

With the predicted $F_{J'}$ and $F_{B'}$, we concatenate them with 2^{P-1}-by-C silhouette feature F_S for representing the human body shape for *GaitSTR*. In addition, we also include the joint feature before refinement as the final representation. The use of a combination of both F_J and $F_{J'}$ ensures that, during training, the network can maximize its ability to distinguish the identities from the skeletons. In addition, using both of the features encoded from the two skeleton sequences gives the most representation for the task of gait recognition.

3.3 Objectives and Inference

We have include two losses for training *GaitSTR*: a triplet loss $L_{triplet}$ for distinguishing the same identities in the same batch and a classification loss L_{cls} for the identities in training set with an MLP layer for projecting the identity feature to the number of candidates. For the combination of the two losses, we follow

$$L = \lambda_1 L_{triplet} + \lambda_2 L_{cls} \quad (4)$$

and empirically set λ_1 as 1. For λ_2 we follow [7], [17] to set it as different values for different datasets. We include further discussion and the choice of parameters in the implementation details section in Section 4.1.

4 Experiments and Results

4.1 Experimental Details

Datasets. In our experiment, we assess our method on four public gait recognition datasets, CASIA-B [15], OUMVLP [1], [24], Gait3D [17] and GREW [18].

CASIA-B [15] has 124 subjects with 10 different walking variants for gait recognition. Among the 10 variants, 6 variants are for normal walking (NM), 2 variants are for the person carrying different bags (BG), and the remaining 2 variants are for different clothes (CL). Each subject has 110 videos captured with 10 variants from 11 different camera viewpoints distributed between 0° and 180°. We follow [5], [6], [7], [29] and use the videos of the first 74 identities for training and the remaining 50 for inference. During inference, we use the first four variances in normal walking conditions (NM) to build the gallery set as the

library to query test sequences. The sequences of the remaining 2 NM variants, along with BG and CL sequences, are used as probe examples for finding the identity in the gallery.

OUMVLP [16], [24] is a large-scale dataset with 10,307 different identities. Each subject in this dataset has 2 different variants for normal walking (NM) conditions from 14 camera viewpoints, making 28 gait sequences. The angles of camera viewpoints are evenly distributed in two bins, 0° to 90°, and 180° to 270°. Every two neighbor viewpoints have a 15-degree gap. We follow [5], [6], [7], [29] to use the identities with odd indexes between the 1-*st* and 10,305-*th* examples and build a training set with 5,153 identities. For the remaining 5,154 identities, we use the first sequence as the gallery set and the second as probes during inference.

Gait3D [17] is a medium dataset compared with CASIA-B and OUMVLP for gait recognition in the wild. It includes 4,000 identities among 25,309 video sequences captured via 39 cameras. Since sequences are captured in the wild, camera positions, carried-on objects, and clothes vary from sequence to sequence. Similar to GREW [18], Gait3D also provides both skeletons and silhouette sequences for each frame in the dataset. We follow [17] to use 3,000 identities for training and the remaining 1,000 during inference. For these 1,000 test cases, we build a probe set with 1,000 sequences for querying, as the probe set, and use the rest 5,369 sequences as the gallery set.

GREW [18] is a large in-the-wild gait recognition dataset with 128,671 sequences capturing 26,345 identities from 882 cameras. Each frame in the video has both silhouettes and poses provided. We follow [18] for using 20,000 identities for training and 6,000 identities as our test set. Each subject in the test set has 4 sequences, where we use two for the gallery and the other two as probe videos following the official split [18].

Implementation Details. For the implementation details section, we will discuss the details for the data preparation, model, and hyperparameter selection in experiments.

Data preparation. For all four datasets, we follow OpenGait[1] to prepare the silhouettes for each dataset, setting the size of each frame to 64×44. Unlike silhouettes, skeletons provided for different datasets vary. Thus, we process the joints for each dataset independently. For the CASIA-B [15] dataset, we follow GaitGraph [8] and use a pretrained HR-Net [48] to generate the skeleton in MS COCO [49] format with 17 joints. The number of frames used for the skeletons of CASIA-B is set to 60, using the 60 frames at the center of the entire sequence as joint input.

For OUMVLP [16] dataset, we follow [24] for applying the skeletons along with the silhouette sequences, and we have skeleton sequences with 18 nodes per frame as OpenPose [38] format. Considering that the sequence length in OUMVLP is shorter than CASIA-B, we set the fixed frame number to 25 for each sequence. For videos shorter than 25, we repeat the frames until we have 25 frames.

For Gait3D [17] and GREW [18], since skeletons are collected in the wild, we normalize each skeleton by setting their height to 2 and move their center to the origin point $(0, 0)$. This can ensure that the position of the skeletons is aligned across different videos and will not change significantly.

In addition to joints, we generate bones based on predefined neighbor link relationships between joints, represented as directional vectors calculated from the differences in coordinates between linked joints. Compared to the joints in the skeletons, the connected bones are defined by neighbor link relationships, emphasizing the numerical connectivity that is not explicitly captured in ST-GCN [13].

Network details. In our network, we have two different encoders. For our silhouette feature encoder, we follow GaitGL [7] to build the encoder for CASIA-B. For Gait3D, OUMVLP, and GREW, we follow GaitBase [23] to encode silhouette features. Note that for GaitMix and GaitRef, we follow [10] to use OpenGait [23] for Gait3D and GaitGL [7] for GREW respectively. For the skeleton feature encoder, we follow ST-GCN [13] for encoding the skeletons into the same embedding dimension N_{out} as the silhouette feature encoder. The dimension of the hidden layers of ST-GCN is set to [64, 64, 128, 128, n_{out}]. The two skeleton decoders of the *GaitSTR* both use the reversed shape of the ST-GCN, with [128, 64, 64, 3] as the hidden dimensions and the number of CMA, K is set to 3. For the encoder and decoder network, we have compared ST-GCN along with other choices, such as MS-G3D [53] for ablation study.

Model training. In our model, we follow [7], [23] for choosing the hyperparameters. For CASIA-B, we use an Adam optimizer [54] with 10^{-4} as the learning rate for 80,000 iterations. We decay the learning rate once at 70,000 iterations for CASIA-B as $\frac{1}{10}$ of its original value. For Gait3D, OUMVLP and GREW, we use the SGD optimizer for 60,000, 120,000 and 180,000 iterations, respectively and set the initial learning rate as 10^{-3}. The learning rate is decayed to $\frac{1}{10}$ three times for these three datasets, at iteration 20,000, 40,000 and 50,000 for Gait3D, 60,000, 80,000 and 100,000 for OUMVLP, and 80,000, 120,000 and 150,000 for GREW. For all four datasets we use, we follow [7], [23] for using 1 for both λ_1 and λ_2 following our ablation results.

Metrics and evaluations. During inference, for each example in the probe set, we use L_2 similarity to find the nearest example in the gallery set. For CASIA-B and OUMVLP, we evaluate the top-1 accuracy for the prediction. For GREW, we evaluate top-1, 5, 10 and 20 accuracies. For Gait3D, we assess top-1 and top-5 accuracies along with mAP and mINP following [55] for assessing since all the correct matches should have low-rank values when pairing the probe example with correct identities in the gallery.

For baseline methods, we compare with state-of-the-art gait recognition approaches, including GaitNet [2], GaitSet [5], GaitPart [6], GLN [29], GaitGL [7], ModelGait [30], and CSTL [50]. Additionally, we compare with PoseGait [33] and GaitGraph [8], which utilize skeleton sequences as their input. For baseline comparison, we include GaitMix and GaitRef [10], which use simple concatenation as described in Section 3.1 and only utilize the skeleton correction network as outlined in Section 3.2, respectively. *GaitSTR*, along with these methods, employs 2-D convolution for binarized silhouette feature extraction. We also include comparisons with other methods which use different modalities, such as GaitEdge [52] and MvModelGait [51].

4.2 Results and Analysis

In this subsection, we first present the numerical results for CASIA-B [15], OUMVLP [1], [24], Gait3D [17], and GREW [18] compared to other state-of-the-art methods. We then delineate and analyze the enhancements conferred by the multi-modal gait model as opposed to the refinement of the skeletons from the silhouettes. In addition, we compare the use of skeletons with 3-D body shapes on the top of silhouettes for gait recognition. Finally, we present an ablation study for our model and visualizations that illustrate the corrected skeletons informed by silhouette guidance.

1. https://github.com/ShiqiYu/OpenGait

TABLE 1
Gait recognition results on CASIA-B dataset, excluding identical-view cases. GaitEdge* requires RGB frames and uses the re-segmented CASIA-B* silhouettes instead of CASIA-B, and MvModelGait requires the input camera viewpoints. TriGait includes a 3-D convolution feature extractor which requires much heavier computation than the 2-D encoders used by other methods in the table. We mark the best results among all the methods in bold and the best results in our baseline methods with underline.

Probe	Method	Camera Positions											Mean
		0°	18°	36°	54°	72°	90°	108°	126°	144°	162°	180°	
NM #5-6	PoseGait [33]	55.3	69.6	73.9	75.0	68.0	68.2	71.1	72.9	76.1	70.4	55.4	68.7
	GaitNet [2]	91.2	92.0	90.5	95.6	86.9	92.6	93.5	96.0	90.9	88.8	89.0	91.6
	GaitGraph [8]	85.3	88.5	91.0	92.5	87.2	86.5	88.4	89.2	87.9	85.9	81.9	87.7
	GaitSet [5]	91.1	98.0	99.6	97.8	95.4	93.8	95.7	97.5	98.1	97.0	88.2	95.6
	GaitPart [6]	94.0	98.7	99.3	98.8	94.8	92.6	96.4	98.3	99.0	97.4	91.2	96.4
	GLN [29]	93.8	98.5	99.2	98.0	95.2	92.9	95.4	98.5	99.0	99.2	91.9	96.5
	GaitGL [7]	95.3	97.9	99.0	97.8	96.1	95.3	97.2	98.9	99.4	98.8	94.5	97.3
	CSTL [50]	97.2	99.0	99.2	98.1	96.2	95.5	97.7	98.7	99.2	98.9	96.5	97.8
	ModelGait [30]	96.9	97.1	98.5	98.4	97.7	98.2	97.6	97.6	98.0	98.4	98.6	<u>97.9</u>
	GaitMix [10]	96.6	98.6	99.2	98.0	97.1	96.2	97.5	98.9	99.3	99.0	94.7	97.7
	GaitRef [10]	97.2	98.7	99.1	98.0	97.3	97.0	98.0	99.4	99.4	98.9	96.4	98.1
	GaitSTR	97.2	98.4	99.2	98.3	97.6	97.8	97.9	99.3	99.3	99.3	97.6	**98.4**
	MvModelGait [51]	97.5	97.6	98.6	98.8	97.7	98.9	98.9	97.3	97.6	97.8	97.9	98.1
	GaitEdge* [52]	97.2	99.1	99.2	98.3	97.3	95.5	97.1	99.4	99.3	98.5	96.4	97.9
	TriGait [25]	97.0	98.6	98.3	98.3	98.4	97.0	98.6	99.0	98.9	98.4	97.4	98.2
BG #1-2	PoseGait [33]	35.3	47.2	52.4	46.9	45.5	43.9	46.1	48.1	49.4	43.6	31.1	44.5
	GaitNet [2]	83.0	87.8	88.3	93.3	82.6	74.8	89.5	91.0	86.1	81.2	85.6	85.7
	GaitGraph [8]	75.8	76.7	75.9	76.1	71.4	73.9	78.0	74.7	75.4	75.4	69.2	74.8
	GaitSet [5]	87.0	93.8	94.6	92.9	88.2	83.0	86.6	92.6	95.7	92.9	83.4	90.1
	GaitPart [6]	89.5	94.5	95.3	93.5	88.5	83.9	89.0	93.6	96.0	94.1	85.3	91.2
	GLN [29]	92.2	95.6	96.7	94.3	91.8	87.8	91.4	95.1	96.3	95.7	87.2	93.1
	GaitGL [7]	93.0	95.7	97.0	95.9	93.3	90.0	91.9	96.8	97.5	96.9	90.7	<u>94.4</u>
	CSTL [50]	91.7	96.5	97.0	95.4	90.9	88.0	91.5	95.8	97.0	95.5	90.3	93.6
	ModelGait [30]	94.8	92.9	93.8	94.5	93.1	92.6	94.0	94.5	89.7	93.6	90.4	93.1
	GaitMix [10]	94.4	96.7	96.8	96.1	94.3	90.4	93.5	97.4	98.0	97.2	92.2	95.2
	GaitRef [10]	94.4	96.4	97.3	96.8	96.2	92.2	94.4	97.2	98.7	97.9	93.3	95.9
	GaitSTR	95.3	97.1	97.8	96.8	96.1	93.2	94.3	96.8	98.3	98.3	94.0	**96.2**
	MvModelGait [51]	93.9	92.5	92.9	94.1	93.4	93.4	95.0	94.7	92.9	93.1	92.1	93.4
	GaitEdge* [52]	95.3	97.4	98.4	97.6	94.3	90.6	93.1	97.8	99.1	98.0	95.0	96.1
	TriGait [25]	91.8	94.3	95.2	96.6	96.5	93.7	95.9	97.6	97.4	96.9	93.8	95.4
CL #1-2	PoseGait [33]	24.3	29.7	41.3	38.8	38.2	38.5	41.6	44.9	42.2	33.4	22.5	36.0
	GaitNet [2]	42.1	58.2	65.1	70.7	68.0	70.6	65.3	69.4	51.5	50.1	36.6	58.9
	GaitGraph [8]	69.6	66.1	68.8	67.2	64.5	62.0	69.5	65.6	65.7	66.1	64.3	66.3
	GaitSet [5]	71.0	82.6	84.0	80.0	71.7	69.1	72.1	76.7	78.5	77.2	63.4	75.1
	GaitPart [6]	72.5	82.8	86.0	82.2	79.5	71.0	77.7	80.8	82.9	81.4	67.7	78.6
	GLN [29]	78.5	90.4	90.3	85.1	80.2	75.8	78.1	81.8	80.9	83.2	72.6	81.5
	GaitGL [7]	71.7	90.5	92.4	89.4	84.9	78.1	83.1	87.5	89.1	83.9	67.4	83.5
	CSTL [50]	78.1	89.4	91.6	86.6	82.1	79.9	81.8	86.3	88.7	86.6	75.3	<u>84.2</u>
	ModelGait [30]	78.2	81.0	82.1	82.8	80.3	76.9	75.5	77.4	72.3	73.5	74.2	77.6
	GaitMix [10]	79.2	89.5	94.2	90.0	84.9	80.3	85.2	89.2	90.3	86.9	73.7	85.8
	GaitRef [10]	81.4	93.3	94.3	91.6	87.8	83.9	88.5	91.7	91.6	89.1	75.0	88.0
	GaitSTR	83.8	94.0	94.9	94.3	90.7	85.5	89.2	91.8	92.8	90.7	78.2	**89.6**
	MvModelGait [51]	77.0	80.0	83.5	86.1	84.5	84.9	80.6	80.4	77.4	76.6	76.9	80.7
	GaitEdge* [52]	84.3	92.8	94.3	92.2	84.6	83.0	83.0	87.5	87.4	85.9	75.0	86.4
	TriGait [25]	91.7	93.2	96.9	97.0	95.2	94.0	94.6	95.3	94.1	94.1	90.8	94.3

Numerical Results. We present our numerical performance on the four datasets used in our experiments in Table 1, 2, 3, and 4, respectively. We follow the official splits of these four datasets for gallery and probe constructions. For CASIA-B and OUMVLP, identical-view cases are excluded.

For all four datasets evaluated, we outperform the existing state-of-the-art methods with *GaitSTR*. In Table 1, on CASIA-B, we achieve the best performance on all splits. Specifically, on NM, BG, and CL, we reduce the error rates from 2.1%, 5.6%, and 15.8% to 1.6%, 3.8%, and 10.4%, respectively, which correspond to a relative reduction of 23.8%, 32.1%, and 34.2% in error rates compared with the best model using 2-D convolution for silhouette feature extraction, while we also show similar performance compared to methods [25] using 3-D convolution. The margin of improvement is even greater for NM and CL settings when compared to our baseline silhouette encoder, GaitGL, where we demonstrate a 33.3% and 37.0% relative reduction for the average rank-1 predictions across all camera views. Furthermore, when compared with GaitEdge [52] and MvModelGait [51], which utilize RGB images and viewpoint angles not typically present in public datasets, *GaitSTR* still exhibits superior performance, indicating the effectiveness of using the skeletons along with silhouettes is able to outperform the methods directly using the RGB images as input, which actually include all the information stored in the silhoeuttes and skeletons.

For the other three datasets, on OUMVLP in Table 2, we show small improvement compared with GaitBase [23] for the top-1 accuracy, while we outperform it along with other methods for all the metrics on Gait3D [17] and GREW [18] in Table 3 and 4, which we show 3.1% and 3.9% improvements on rank-1 accuracies respectively compared with other state-of-the-art methods using 2-D convolution as silhouette feature encoders. We also show improvement on other metrics, such as mAP and mINP [17] used by Gait3D [17].

TABLE 2
Gait recognition results for accuracy across all the test views on OUMVLP dataset, excluding identical-view cases.

Method	Camera Positions														Mean
	0°	15°	30°	45°	60°	75°	90°	180°	195°	210°	225°	240°	255°	270°	
GEINet [56]	23.2	38.1	48.0	51.8	47.5	48.1	43.8	27.3	37.9	46.8	49.9	45.9	45.7	41.0	42.5
GaitSet [5]	79.2	87.7	89.9	90.1	87.9	88.6	87.7	81.7	86.4	89.0	89.2	87.2	87.7	86.2	87.0
GaitPart [6]	82.8	89.2	90.9	91.0	89.7	89.9	89.3	85.1	87.7	90.0	90.1	89.0	89.0	88.1	88.7
GLN [29]	83.8	90.0	91.0	91.2	90.3	90.0	89.4	85.3	89.1	90.5	90.6	89.6	89.3	88.5	89.2
GaitGL [7]	84.2	89.8	91.3	91.7	90.8	91.0	90.4	88.1	88.2	90.5	90.5	89.5	89.7	88.8	89.6
MvModelGait [51]	87.7	89.7	91.1	90.1	89.8	90.3	90.3	88.1	89.4	89.4	90.0	90.8	90.0	89.7	89.7
CSTL [50]	87.1	91.0	91.5	91.8	90.6	90.8	90.6	89.4	90.2	90.5	90.7	89.8	90.0	89.4	90.2
GaitBase [23]	87.2	91.2	91.8	92.0	91.4	91.2	90.8	88.9	90.4	91.1	91.3	90.7	90.5	90.0	90.6
GaitMix [10]	85.4	90.3	91.2	91.5	91.2	90.9	90.5	88.9	88.7	90.3	90.5	89.8	89.6	88.9	89.9
GaitRef [10]	85.7	90.5	91.6	91.9	91.3	91.3	90.9	89.3	89.0	90.8	90.8	90.1	90.1	89.5	90.2
GaitSTR	87.6	91.5	91.8	92.1	91.5	91.3	91.0	89.2	90.7	91.1	91.3	90.8	90.6	90.2	**90.8**

TABLE 3
Gait recognition results reported on the Gait3D dataset with 64×44 as input sizes. For all four metrics, higher values of the same metric indicate better performance. B represents the bone input.

Methods	Rank@1	Rank@5	mAP	mINP
GaitSet [5]	36.70	58.30	30.01	17.30
GaitPart [6]	28.20	47.60	21.58	12.36
GLN [29]	31.40	52.90	24.74	13.58
GaitGL [7]	29.70	48.50	22.29	13.26
OpenGait [17]	42.90	63.90	35.19	20.83
CSTL [50]	11.70	19.20	5.59	2.59
GaitBase [23]	62.00	78.80	53.17	35.33
SMPLGait [17]	46.30	64.50	37.16	22.23
GaitMix [10]	46.20	66.20	37.08	22.85
GaitRef [10]	49.00	69.30	40.69	25.26
GaitSTR	**65.10**	**81.30**	**55.59**	**36.84**

TABLE 4
Rank-1, 5, 10 and 20 accuracies on GREW dataset.

Methods	Rank-1	Rank-5	Rank-10	Rank-20
PoseGait [33]	0.2	1.1	2.2	4.3
GaitGraph [8]	1.3	3.5	5.1	7.5
GEINet [56]	6.8	13.4	17.0	21.0
TS-CNN [3]	13.6	24.6	30.2	37.0
GaitSet [5]	46.3	63.6	70.3	76.8
GaitPart [6]	44.0	60.7	67.4	73.5
CSTL [50]	50.6	65.9	71.9	76.9
GaitGL [7]	51.4	67.5	72.8	77.3
GaitBase [23]	60.1	75.5	80.4	84.1
GaitMix [10]	52.4	67.4	72.9	77.2
GaitRef [10]	53.0	67.9	73.0	77.5
GaitSTR	**64.0**	**78.5**	**83.2**	**86.3**

TABLE 5
Rank-1 accuracy of the variations skeletons in addition to silhouettes for gait recognition on CASIA-B. 'Sils.' represents silhouettes.

Input Modality	Methods	NM	CL	BG	Params
Sil. only	GaitGL [7]	97.3	94.4	83.5	3.82M
Sil. + Joint	GaitMix [10]	97.7	95.2	85.8	3.84M
	GaitRef [10]	98.1	95.9	88.0	8.40M
Sil. + Joint + Bone	GaitMix (w/ Bone)	98.0	95.6	87.5	-
	GaitRef (w/ Bone)	98.2	96.0	88.9	-
	+ CMA$_{B \to J}$	98.1	95.7	88.7	-
	GaitSTR	**98.4**	**96.2**	**89.6**	11.48M

Improvement of Skeleton Refinement. With the inclusion of both bones and joints to represent skeletons, we further analyze the improvement from introducing a new modality and the manner in which these modalities are utilized in *GaitSTR*. We present the results in Table 5, showing rank-1 accuracy on CASIA-B [15]. We begin with our baseline method, GaitGL [7], which operates on a single modality, and then proceed to analyze the introduction of joints and bones, as well as various ways of integrating them with silhouettes for different combinations. As our model includes an extra refinement model for skeleton correction, we also include the corresponding number of parameters used in the network. Even with the increase in the number of parameters required, the inference time for a 30-frame gait sequence is about 45× faster than real-time on a single Nvidia 3090 Ti GPU.

When comparing the use of silhouettes as the only input for gait recognition, the introduction of joints as skeletons displays an improvement for both simple feature correction, as seen with GaitMix [10], and the use of silhouettes to refine joints, as with GaitRef [10] across all three metrics. GaitRef, which uses silhouettes to refine the joints, provides better recognition accuracy compared to simply aggregating the two features.

Furthermore, the introduction of bones in both GaitMix and GaitRef leads to additional improvements over the use of joints alone with silhouettes. For GaitRef, we treat the newly introduced bones similarly to joints and utilize silhouette features to refine them. We then concatenate the three encoded features (silhouettes, joints, and bones) for recognition. Including a single-sided bone-to-joint cross-modal adapter results in a slight decrease in performance, whereas incorporating adapters on both sides, as in *GaitSTR*, shows better performance. The refinement from one side creates inconsistencies across the two skeletal representations, while the two-way sequential refinement provides consistent enhancement across these modalities.

Skeleton and Body Shape for Gait Recognition. In addition to silhouettes and skeletons, Gait3D [17] provides 3-D body shapes alongside silhouette sequences, which are utilized by SMPLGait [17]. In Table 3, we provide a comparison of using 3-D body shapes as in SMPLGait [17] and using skeletons by removing the skeleton correction network and cross-modal adapters, which directly aggregate skeleton features with joint features as in GaitMix [10] and GaitRef [10]. For a fair comparison, we use GaitMix and GaitRef [10] with the OpenGait [23] baseline without augmentation, as SMPLGait [18] is not implemented on the latest OpenGait configuration with data augmentation.

TABLE 6
Ablation results for different silhouette and skeleton feature combination on CASIA-B dataset for three splits. 'Padding' indicates the skeleton feature is padded on each of the feature of different scales, while 'concat.' means we concatenate the feature along with the scale dimension and use it only once.

Method	Combination	NM	CL	BG
GaitGL [7]	N/A	97.3	94.4	83.5
Sil. + Joints	Padding	97.5	94.6	85.8
Sil. + Joints	Concat.	**98.1**	**95.9**	**88.0**

TABLE 7
Ablations for different encoder and decoder combinations for silhouette with joints and different skeleton smoothing methods on CASIA-B datasets. Results are reported in Top-1 accuarcy.

Encoder	Decoder	NM	CL	BG
ST-GCN	N/A	97.7	95.2	85.8
MS-G3D	N/A	98.0	95.5	86.4
ST-GCN	ST-GCN	**98.1**	**95.9**	88.0
ST-GCN	MS-G3D	**98.1**	95.7	**88.5**
MS-G3D	ST-GCN	**98.1**	**95.9**	88.3
Average Smoothing		97.6	95.0	85.6
Gaussian Smoothing		97.7	95.2	85.9
SmoothNet [9]		97.4	94.4	83.8

TABLE 8
Ablation results of different input for the skeleton correction network on CASIA-B. SCN is skeleton correction network.

Corr. Input	NM	BG	CL
w/o F_J	97.7	95.4	87.0
w/o F_S	97.6	95.3	85.6
w/o J	97.3	95.5	86.0
Full SCN	**98.1**	**95.9**	**88.0**

TABLE 9
Ablation results for λ_1, λ_2 and number of frames on CASIA-B.

λ_1	λ_2	# Frame	NM	CL	BG	Avg.
1	1	30	98.4	96.2	89.6	**94.7**
1	1	20	97.9	95.3	86.9	93.4
1	1	10	94.7	90.0	70.8	95.2
0.01	1	30	96.7	92.2	76.5	88.5
0.1	1	30	97.5	94.2	82.5	91.4
10	1	30	98.0	95.8	90.1	94.6
1	0.01	30	98.2	95.8	89.5	94.5
1	0.1	30	98.2	96.0	89.5	94.6
1	10	30	97.7	94.5	83.7	92.0

Compared to using silhouettes as the only input modality, the inclusion of skeletons and body shapes both enhance recognition accuracy. In SMPLGait, skeleton information is partially integrated into the generated 3-D body shapes for gait recognition, making SMPLGait yields similar performance as GaitMix [10], which includes joints and bones, across all four metrics.

When compared to SMPLGait, which uses a 3-D body shape as the second modality, GaitRef [10] with bone inputs for the refined skeletons achieves better recognition performance. Considering that the generation of SMPL body shapes also requires skeletons [57], inaccurate pose estimation in 3-D body shape generation can hinder the model's ability to correctly interpret noisy body shapes with erroneous poses in SMPLGait [17]. GaitRef, however, does not suffer from this issue with refined skeletons.

Ablation Studies. For ablation studies, we present results on: 1) different methods of combining skeleton and silhouette features, 2) various skeleton encoder and decoder networks in comparison with other skeleton refinement methods, and 3) the inputs of the skeleton correction network. All experiments were conducted on the CASIA-B dataset [15] for each of the three different settings, and we present the Top-1 accuracy for the final gait recognition results. The results are shown in Tables 6, 7, and 8, respectively. Since the joint and bones branches are identical and exhibit similar performance, our ablation experiments focus on the joints branch as skeleton representation.

(i) Feature Combination. In addition to concatenating the features, we also repeat and pad the skeleton feature along with each segment of the silhouette features to provide the guidance for different level of silhouette embeddings, which we label as 'padding'. We show the results in Table 6. For comparison, we also add the performance of GaitGL [7] in the table, which only uses the silhouette feature for gait recognition and is our backbone baseline on CASIA-B. We observe that padding the skeleton feature alongside each size of the silhouette feature results in worse performance compared to concatenating the refined feature just once as the final recognition feature. Padding the skeleton feature multiple times may cause the skeleton input to dominate the feature space, whereas concatenating it once allows the silhouette features to contribute robust information about the pose and remain less sensitive to skeletons.

(ii) Encoder-decoder Variations. For the choice of the skeleton encoder and skeleton correction decoder, we select between two state-of-the-art skeleton action recognition models: ST-GCN [13] and MS-G3D [53]. The results are presented in Table 7. Following previous observations, where using joints as the skeleton representation follows the same trend as using both joints and bones, we opt for using only joints as the skeleton representation for this ablation. Both MS-G3D and ST-GCN show improvement in performance. However, in our experiments, MS-G3D requires significantly more GPU memory and at least double the training time for each module introduced. Considering their comparable performance and time efficiency, we choose ST-GCN for both the encoder and decoder in our final pipeline.

(iii) Skeleton Refinement. For skeleton refinement, we compare the refining of the skeleton sequence using silhouettes with neighbor smoothing (average and Gaussian window for the neighboring three frames) and SmoothNet [9] (pretrained on H36m [58]) on the CASIA-B dataset based on Top-1 accuracy. The results, displayed in Table 7, demonstrate that refining skeleton with silhouettes outperforms the other methods. Among the three variations, 3-frame Gaussian smoothing shows a slight improvement but still falls short compared to using silhouettes.

Different from naive temporal smoothing, which can result in poses inconsistent in the sequence, integrating silhouette features introduces walking patterns not present in the skeletons, aiding their self-refinement for gait recognition. Compared to skeletons refined from the skeleton sequence alone, external knowledge from encoded silhouette embeddings reduces ambiguity, providing

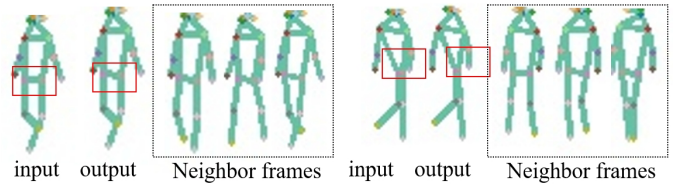

Fig. 4. Visualization of successful and failed refined skeletons with *GaitSTR*. For each example, from left to right, we have original skeletons, refined skeletons and its neighbor frames.

ID-specific information during training when the walking pattern cannot be correctly extracted from the skeleton alone.

(iv) Input of the Skeleton Correction Network. Considering three distinct inputs in our skeleton correction network, F_J, F_S, and J, we investigate each component's contributions and present the results in Table 8 using three splits of the CASIA-B datasets. We note that when either of the three input is excluded, there is a significant drop in performance. The skeleton correction network capitalizes on temporal consistency in the skeleton sequences for correction, while the additional silhouette information provides external support for an enhanced understanding.

(v) Weights for losses and number of frames. As we set λ_1 and λ_2 to be 1 in our main experiments, we include ablation results for different λ combinations on CASIA-B in Table 4.2, along with the number of frames of videos used in our experiment. We note that setting both λ_1 and λ_2 to 1 shows the best average performance. In addition, with more frames available, models are able to perform better compared with fewer frames available.

Skeletons Visualization. We present two examples from *GaitSTR* compared to the original skeletons in Figure 4, accompanied by the three nearest silhouettes from a similar timestamp. With two modalities of representations for gait, *GaitSTR* can make more precise modifications to the skeletons' nodes. However, it still fails on some obvious errors, as seen in the second example in Figure 4.

5 Conclusion

We introduce *GaitSTR*, building on GaitMix and GaitRef [10], to integrate and refine skeletons with silhouettes for gait recognition. *GaitSTR* incorporates bone representation alongside the joints, emphasizing the numerical connectivities between different nodes. It combines silhouettes and skeletons with two levels of refinement: silhouette-to-skeleton refinement for general guidance and dual-layer cross-modal adapters for sequential two-stream refinement between the joints and bones, ensuring temporal consistency across different representations. We compare *GaitSTR* on four public datasets, including CASIA-B, OUMVLP, Gait3D, and GREW, and demonstrate state-of-the-art performance compare with other gait recognition methods.

Acknowledgments

This research is based upon work supported in part by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), via [2022-21102100007]. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of ODNI, IARPA, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright annotation therein.

References

[1] Y. He, J. Zhang, H. Shan, and L. Wang, "Multi-task gans for view-specific feature learning in gait recognition," *TIFS*, vol. 14, no. 1, pp. 102–113, 2018. 1, 4, 5

[2] C. Song, Y. Huang, Y. Huang, N. Jia, and L. Wang, "Gaitnet: An end-to-end network for gait based human identification," *PR*, vol. 96, p. 106988, 2019. 1, 5, 6

[3] Z. Wu, Y. Huang, L. Wang, X. Wang, and T. Tan, "A comprehensive study on cross-view gait based human identification with deep cnns," *TPAMI*, vol. 39, no. 2, pp. 209–226, 2016. 1, 7

[4] S. Yu, H. Chen, Q. Wang, L. Shen, and Y. Huang, "Invariant feature extraction for gait recognition using only one uniform model," *Neurocomputing*, vol. 239, pp. 81–93, 2017. 1

[5] H. Chao, Y. He, J. Zhang, and J. Feng, "Gaitset: Regarding gait as a set for cross-view gait recognition," in *AAAI*, 2019, pp. 8126–8133. 1, 2, 4, 5, 6, 7

[6] C. Fan, Y. Peng, C. Cao, X. Liu, S. Hou, J. Chi, Y. Huang, Q. Li, and Z. He, "Gaitpart: Temporal part-based model for gait recognition," in *CVPR*, 2020, pp. 14 225–14 233. 1, 2, 4, 5, 6, 7

[7] B. Lin, S. Zhang, and X. Yu, "Gait recognition via effective global-local feature representation and local temporal aggregation," in *ICCV*, 2021, pp. 14 648–14 656. 1, 2, 4, 5, 6, 7, 8

[8] T. Teepe, A. Khan, J. Gilg, F. Herzog, S. Hörmann, and G. Rigoll, "Gaitgraph: Graph convolutional network for skeleton-based gait recognition," in *ICIP*, 2021, pp. 2314–2318. 1, 2, 5, 6, 7

[9] A. Zeng, L. Yang, X. Ju, J. Li, J. Wang, and Q. Xu, "Smoothnet: A plug-and-play network for refining human poses in videos," *ECCV*, 2022. 1, 2, 8

[10] H. Zhu, W. Zheng, Z. Zheng, and R. Nevatia, "Gaitref: Gait recognition with refined sequential skeletons," *IJCB*, 2023. 1, 2, 5, 6, 7, 8, 9

[11] L. Shi, Y. Zhang, J. Cheng, and H. Lu, "Two-stream adaptive graph convolutional networks for skeleton-based action recognition," in *CVPR*, 2019, pp. 12 026–12 035. 1

[12] H. Zhu, Z. Zheng, and R. Nevatia, "Temporal shift and attention modules for graphical skeleton action recognition," in *ICPR*, 2022, pp. 3145–3151. 1, 2

[13] S. Yan, Y. Xiong, and D. Lin, "Spatial temporal graph convolutional networks for skeleton-based action recognition," in *AAAI*, 2018. 1, 3, 5, 8

[14] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *CVPR*, 2016, pp. 770–778. 2

[15] S. Yu, D. Tan, and T. Tan, "A framework for evaluating the effect of view angle, clothing and carrying condition on gait recognition," in *ICPR*, vol. 4, 2006, pp. 441–444. 2, 4, 5, 7, 8

[16] N. Takemura, Y. Makihara, D. Muramatsu, T. Echigo, and Y. Yagi, "Multi-view large population gait dataset and its performance evaluation for cross-view gait recognition," *TCVA*, vol. 10, no. 1, pp. 1–14, 2018. 2, 5

[17] J. Zheng, X. Liu, W. Liu, L. He, C. Yan, and T. Mei, "Gait recognition in the wild with dense 3d representations and a benchmark," in *CVPR*, 2022, pp. 20 228–20 237. 2, 4, 5, 6, 7, 8

[18] Z. Zhu, X. Guo, T. Yang, J. Huang, J. Deng, G. Huang, D. Du, J. Lu, and J. Zhou, "Gait recognition in the wild: A benchmark," in *ICCV*, 2021, pp. 14 789–14 799. 2, 4, 5, 6, 7

[19] H. Dou, P. Zhang, W. Su, Y. Yu, Y. Lin, and X. Li, "Gaitgci: Generative counterfactual intervention for gait recognition," in *CVPR*, 2023, pp. 5578–5588. 2

[20] M. Wang, X. Guo, B. Lin, T. Yang, Z. Zhu, L. Li, S. Zhang, and X. Yu, "Dygait: Exploiting dynamic representations for high-performance gait recognition," *ICCV*, 2023. 2

[21] K. Ma, Y. Fu, D. Zheng, C. Cao, X. Hu, and Y. Huang, "Dynamic aggregated network for gait recognition," in *CVPR*, 2023, pp. 22 076–22 085. 2

[22] Z. Huang, D. Xue, X. Shen, X. Tian, H. Li, J. Huang, and X.-S. Hua, "3d local convolutional neural networks for gait recognition," in *ICCV*, 2021, pp. 14 920–14 929. 2

[23] C. Fan, J. Liang, C. Shen, S. Hou, Y. Huang, and S. Yu, "Opengait: Revisiting gait recognition toward better practicality," 2023. 2, 5, 6, 7

[24] W. An, S. Yu, Y. Makihara, X. Wu, C. Xu, Y. Yu, R. Liao, and Y. Yagi, "Performance evaluation of model-based gait on multi-view very large population database with pose sequences," *TBIOM*, vol. 2, no. 4, pp. 421–430, 2020. 2, 4, 5

[25] Y. Sun, X. Feng, L. Ma, L. Hu, and M. Nixon, "Trigait: Aligning and fusing skeleton and silhouette gait data via a tri-branch network," in *IJCB*, 2023. 2, 6
[26] H. Guo and Q. Ji, "Physics-augmented autoencoder for 3d skeleton-based gait recognition," in *ICCV*, 2023, pp. 19 627–19 638. 2
[27] X. Huang, X. Wang, Z. Jin, B. Yang, B. He, B. Feng, and W. Liu, "Condition-adaptive graph convolution learning for skeleton-based gait recognition," *TIP*, 2023. 2
[28] Y. Cui and Y. Kang, "Multi-modal gait recognition via effective spatial-temporal feature fusion," in *CVPR*, 2023, pp. 17 949–17 957. 2
[29] S. Hou, C. Cao, X. Liu, and Y. Huang, "Gait lateral network: Learning discriminative and compact representations for gait recognition," in *ECCV*, 2020, pp. 382–398. 2, 4, 5, 6, 7
[30] X. Li, Y. Makihara, C. Xu, Y. Yagi, S. Yu, and M. Ren, "End-to-end model-based gait recognition," in *ACCV*, 2020. 2, 5, 6
[31] H. Zhu, W. Zheng, Z. Zheng, and R. Nevatia, "Sharc: Shape and appearance recognition for person identification in-the-wild," *arXiv preprint arXiv:2310.15946*, 2023. 2
[32] H. Zhu, Z. Zheng, and R. Nevatia, "Gait recognition using 3-d human body shape inference," in *WACV*, 2023, pp. 909–918. 2
[33] R. Liao, S. Yu, W. An, and Y. Huang, "A model-based gait recognition method with body pose and human prior knowledge," *PR*, 2020. 2, 5, 6, 7
[34] J. Wang, K. Sun, T. Cheng, B. Jiang, C. Deng, Y. Zhao, D. Liu, Y. Mu, M. Tan, X. Wang *et al.*, "Deep high-resolution representation learning for visual recognition," *TPAMI*, vol. 43, no. 10, pp. 3349–3364, 2020. 2
[35] L. Wang, R. Han, J. Chen, and W. Feng, "Combining the silhouette and skeleton data for gait recognition," *arXiv preprint arXiv:2202.10645*, 2022. 2
[36] E. Pinyoanuntapong, A. Ali, P. Wang, M. Lee, and C. Chen, "Gaitmixer: skeleton-based gait representation learning via wide-spectrum multi-axial mixer," in *ICASSP*, 2023, pp. 1–5. 2
[37] B. Xiao, H. Wu, and Y. Wei, "Simple baselines for human pose estimation and tracking," in *ECCV*, 2018. 2
[38] Z. Cao, G. Hidalgo, T. Simon, S.-E. Wei, and Y. Sheikh, "Openpose: realtime multi-person 2d pose estimation using part affinity fields," *TPAMI*, vol. 43, no. 1, pp. 172–186, 2019. 2, 5
[39] K. Li, S. Wang, X. Zhang, Y. Xu, W. Xu, and Z. Tu, "Pose recognition with cascade transformers," in *CVPR*, 2021, pp. 1944–1953. 2
[40] Y. Li, S. Zhang, Z. Wang, S. Yang, W. Yang, S.-T. Xia, and E. Zhou, "Tokenpose: Learning keypoint tokens for human pose estimation," in *ICCV*, 2021. 2
[41] S. Yang, Z. Quan, M. Nie, and W. Yang, "Transpose: Keypoint localization via transformer," in *ICCV*, 2021. 2
[42] Y. Xu, J. Zhang, Q. Zhang, and D. Tao, "Vitpose: Simple vision transformer baselines for human pose estimation," *arXiv preprint arXiv:2204.12484*, 2022. 2
[43] Y. Yuan, R. Fu, L. Huang, W. Lin, C. Zhang, X. Chen, and J. Wang, "Hrformer: High-resolution transformer for dense prediction," in *NeurIPS*, 2021. 2
[44] D. Rempe, T. Birdal, A. Hertzmann, J. Yang, S. Sridhar, and L. J. Guibas, "Humor: 3d human motion model for robust pose estimation," in *ICCV*, 2021, pp. 11 488–11 499. 2
[45] H. Xu, Y. Gao, Z. Hui, J. Li, and X. Gao, "Language knowledge-assisted representation learning for skeleton-based action recognition," *arXiv preprint arXiv:2305.12398*, 2023. 2
[46] Y. Fu, Y. Wei, Y. Zhou, H. Shi, G. Huang, X. Wang, Z. Yao, and T. Huang, "Horizontal pyramid matching for person re-identification," in *AAAI*, vol. 33, 2019, pp. 8295–8302. 3
[47] H. Zhu, Y. Yuan, Y. Zhu, X. Yang, and R. Nevatia, "Open: Order-preserving pointcloud encoder decoder network for body shape refinement," in *ICPR*, 2022. 4
[48] K. Sun, B. Xiao, D. Liu, and J. Wang, "Deep high-resolution representation learning for human pose estimation," in *CVPR*, 2019, pp. 5693–5703. 5
[49] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *ECCV*, 2014, pp. 740–755. 5
[50] X. Huang, D. Zhu, H. Wang, X. Wang, B. Yang, B. He, W. Liu, and B. Feng, "Context-sensitive temporal feature learning for gait recognition," in *ICCV*, 2021, pp. 12 909–12 918. 5, 6, 7
[51] X. Li, Y. Makihara, C. Xu, and Y. Yagi, "End-to-end model-based gait recognition using synchronized multi-view pose constraint," in *ICCV*, 2021, pp. 4106–4115. 5, 6, 7
[52] J. Liang, C. Fan, S. Hou, C. Shen, Y. Huang, and S. Yu, "Gaitedge: Beyond plain end-to-end gait recognition for better practicality," *arXiv preprint arXiv:2203.03972*, 2022. 5, 6
[53] Z. Liu, H. Zhang, Z. Chen, Z. Wang, and W. Ouyang, "Disentangling and unifying graph convolutions for skeleton-based action recognition," in *CVPR*, 2020, pp. 143–152. 5, 8
[54] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014. 5
[55] M. Ye, J. Shen, G. Lin, T. Xiang, L. Shao, and S. C. Hoi, "Deep learning for person re-identification: A survey and outlook," *TPAMI*, vol. 44, no. 6, pp. 2872–2893, 2021. 5
[56] K. Shiraga, Y. Makihara, D. Muramatsu, T. Echigo, and Y. Yagi, "Geinet: View-invariant gait recognition using a convolutional neural network," in *ICB*, 2016, pp. 1–8. 7
[57] Y. Sun, W. Liu, Q. Bao, Y. Fu, T. Mei, and M. J. Black, "Putting people in their place: Monocular regression of 3d people in depth," in *CVPR*, 2022, pp. 13 243–13 252. 8
[58] C. Ionescu, D. Papava, V. Olaru, and C. Sminchisescu, "Human3. 6m: Large scale datasets and predictive methods for 3d human sensing in natural environments," *TPAMI*, pp. 1325–1339, 2013. 8

Wanrong Zheng received her MS degree in Computer Science from the University of Southern California (USC) in 2023. Before her time at USC, she worked at SenseTime as a full-time research engineer from 2019 to 2021. Her research interests are computer vision and deep learning, focusing on multi-modal perception and explainable artificial intelligence.

Haidong Zhu received the BS degree in Electronic Information Science and Technology from Tsinghua University in 2019. He is currently pursuing the Ph.D. degree with the Department of Computer Science, University of Southern California. His interests include computer vision, focus on 3-D vision and its application for downstream vision tasks.

Zhaoheng Zheng is currently pursuing his Ph.D. in the Computer Science Department, University of Southern California. His research interest includes joint understanding of vision-langauge corpora and large multimodal models. Prior to USC, he obtained his M.S. in Computer Science from University of Michigan, Ann Arbor and his B.Eng. in Computer Science and Technology from Tsinghua University.

Ram Nevatia (Fellow, IEEE) received his PhD degree in Electrical Engineering from Stanford University, California. He has been with the University of Southern California since 1975, where he is currently Fletcher Jones Professor of computer science and Professor of electrical and computer engineering. He has made contributions to various areas of computer vision including object recognition, image grounding and action recognition.