DIFFUSION PRETRAINING FOR GAIT RECOGNITION IN THE WILD

Supplementary Materials

Table 1. Reproduced baseline results on Gait3D. mAP and mINP denote mean average precision and mean inverse negative penalty, respectively.

Model	Gait3D					
	✗ Data Aug.			✓ Data Aug.		
	Rank-1	mAP	mINP	Rank-1	mAP	mINP
Euclidean Distance						
GaitGL	28.1	20.4	12.2	31.5	22.4	13.2
GaitPart	26.6	20.1	11.8	29.7	22.0	12.8
GaitSet	37.9	30.4	17.9	39.7	32.0	19.0
SMPLGait w/o 3D	44.8	35.7	21.6	42.9	33.4	19.5
GaitBase	55.5	46.6	30.1	63.2	53.2	35.1
Cosine Distance						
GaitGL	29.2	21.1	12.3	32.4	22.9	13.7
GaitPart	31.2	23.2	13.5	38.7	29.4	17.6
GaitSet	42.2	33.4	19.4	47.8	39.4	24.4
SMPLGait w/o 3D	45.5	36.8	21.7	42.9	33.7	20.0
GaitBase	56.5	47.3	29.1	65.8	55.8	36.5

Table 2. Reproduced baseline results on GREW.

Model	GREW					
	✗ Data Aug.			✓ Data Aug.		
	Rank-1	Rank-5	Rank-10	Rank-1	Rank-5	Rank-10
Euclidean Distance						
GaitGL	50.8	66.7	72.5	56.3	71.2	76.0
GaitSet	47.5	64.7	71.2	51.7	68.1	74.0
SMPLGait w/o 3D	45.8	63.3	70.0	50.9	68.6	74.9
GaitBase	58.9	73.7	79.0	59.4	74.3	79.9
Cosine Distance						
GaitGL	54.0	68.8	73.9	58.4	72.3	77.2
GaitSet	48.1	65.2	71.4	53.1	69.7	75.7
SMPLGait w/o 3D	47.6	64.8	71.2	52.1	69.0	74.8
GaitBase	58.1	73.5	78.6	61.8	76.7	81.7

1. EXPERIMENTS

1.1. Reproduction of Baseline Results

Using the OpenGait [1] framework, we reproduced five existing deep gait recognition models— GaitGL [2], GaitPart [3], GaitSet [4], SMPLGait w/o 3D [5], and GaitBase [1]—both with and without data augmentation during training. Each model was trained using the standard supervised learning objective and evaluated for its gait recognition performance. Table 1 and Table 2 present the reproduced baseline results on Gait3D [5] and GREW [6], respectively. Since we primarily used a single GPU to train each model, rather than the multi-GPU settings used in OpenGait, our reproduced Euclidean baseline results differ slightly from those reported in the OpenGait repository. Nonetheless, the results are similar.

Based on our results, we see that training with cosine distance and Euclidean distance yielded significantly different gait recognition performances. Specifically, cosine distance consistently outperformed Euclidean distance on both the Gait3D and GREW datasets. Consequently, we opted to use cosine distance for both training and evaluation throughout our study. As training for GaitPart is unstable on the GREW dataset when cosine distance is used, we decided to exclude GaitPart from our applicability study on GREW.

1.2. Hyperparameter Settings

The default hyperparameters used during diffusion pretraining are listed in Table 3. Depending on the memory requirement, we used either RTX 3090 or RTX A6000 GPUs for pretraining, as indicated in Table 4. To ensure fairness, both the reproduction of the baseline results in Section 1.1 and our transfer learning experiments were conducted using the same GPU shown in Table 5. For GaitBase with data augmentation, as it was impossible to achieve similar performance as stated in OpenGait using a single GPU, we had to turn to use the multi-GPU settings originally used by the authors.

1.3. Complete Pretraining Findings

Fig. 1 and Fig. 2 show the gait recognition performance for the different gait feature extractors used at different checkpoints of the pretraining process with the Gait3D and GREW datasets, respectively. For all cases, we observe a steady improvement in gait recognition performance during the diffusion pretraining process.

Fig. 3 and Fig. 4 show the recorded mean cosine distance of the gait features of anchor-positive pairs and the gait features of anchor-negative pairs within a batch during pretraining on the Gait3D and GREW datasets, respectively, for the different gait feature extractors used. We see that regardless of the gait feature extractor used, the difference in the cosine distances between the anchor-positive pairs and anchor-negative pairs increases and stabilises during diffusion pretraining.

1.4. Transfer Learning Results with Different r Values

Table 6 shows the rank-1 accuracy of the various models on the Gait3D and GREW datasets with different learning rate ratio between pretrained and untrained layers, r, attempted.

Table 3. Default hyperparameters used during diffusion pretraining.

Hyperparameter	Value
Latent Diffusion Settings	
Autoencoder	Tiny AutoEncoder for Stable Diffusion
Diffusion Timestep	1000
Noise Scheduler	Inverted Cosine
Loss-Weighting Strategy	Medium-Level Noise Prioritization
Condition Pooling Method	Mean
Video Diffusion Model Settings	
Initial Kernel Size	5
Initial Dimension	64
Dimension Multiplier	[1, 2, 4]
Number of Attention Heads	8
Attention Dimension (Per Head)	32
Number of Input Frames	30
Timestep Condition Dimension	256
GroupNorm Number of Groups	32
Pretraining Settings	
Batch Size	64 (Gait3D), 128 (GREW)
Optimizer	AdamW
Learning Rate (Denoiser)	$1e^{-4}$
Learning Rate (Gait Feature Extractor)	$5e^{-4}$
Learning Rate Scheduler	CosineAnnealingLR
Training Iterations	120000
Warmup Iterations	2000
Warmup Start Factor	0.01
Input Data Augmentation Settings	
RandomAffine Probability	0.2
RandomPerspective Probability	0.2
RandomHorizontalFlip Probability	0.5
RandomPartDilate Probability	0.2

Table 4. GPU used during diffusion pretraining.

Method	Gait3D	GREW
GaitGL	RTX 3090	RTX A6000
GaitPart	RTX 3090	-
GaitSet	RTX 3090	RTX 3090
SMPLGait w/o 3D	RTX 3090	RTX 3090
GaitBase	RTX 3090	RTX A6000

Table 5. GPU used during reproduction of baseline and transfer learning.

Method	Gait3D		GREW	
	✗ Data Aug.	✓ Data Aug.	✗ Data Aug.	✓ Data Aug.
GaitGL	RTX 3090	RTX 3090	RTX A6000	RTX A6000
GaitPart	RTX 3090	RTX 3090	-	-
GaitSet	RTX 3090	RTX 3090	RTX 3090	RTX 3090
SMPLGait w/o 3D	RTX 3090	RTX 3090	RTX 3090	RTX 3090
GaitBase	RTX 3090	RTX A6000	RTX A6000	4 × A100

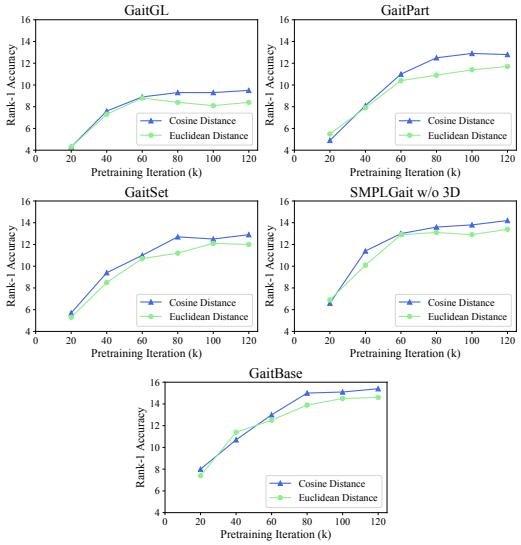

Fig. 1. Rank-1 accuracy curves during diffusion pretraining on Gait3D.

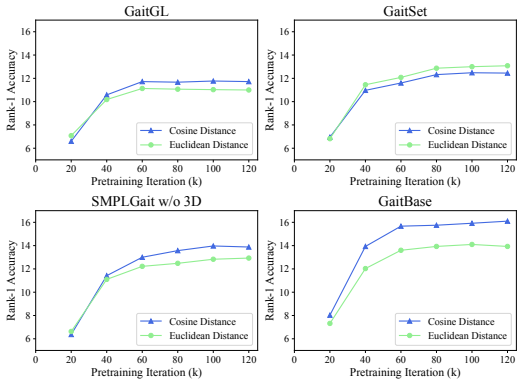

Fig. 2. Rank-1 accuracy curves during diffusion pretraining on GREW.

2. ABLATION STUDIES

In this section, we present the various ablation studies conducted to investigate the effects that different hyperparameter settings have on the diffusion pretraining process and downstream gait recognition tasks. For all experiments, we used the backbone of SMPLGait w/o 3D as our gait feature extractor and Gait3D as the dataset. During transfer learning, no data augmentation was applied, and the pretrained backbone was finetuned with a lower learning rate ($0.1\times$) than the untrained layers. Aside from the hyperparameter being investigated, all other hyperparameters were assigned based on the default pretraining and transfer learning settings. The results of the various ablation studies are summarized in Table 7.

Noise Scheduler and Loss Weighting: We explored two kinds of schedulers—a typical cosine scheduler [7] and an inverted cosine scheduler [8]. For the cosine scheduler, we at-

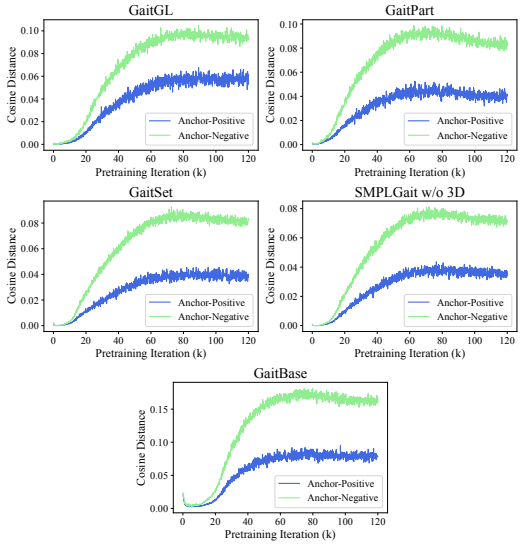

Fig. 3. Mean cosine distance of anchor-positive pair and anchor-negative pair during diffusion pretraining on Gait3D.

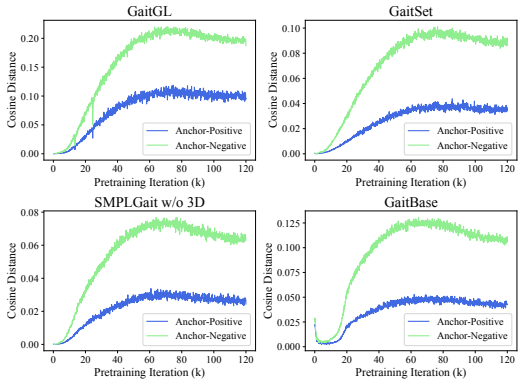

Fig. 4. Mean cosine distance of anchor-positive pair and anchor-negative pair during diffusion pretraining on GREW.

tempted a uniform weighting strategy, where losses from different timesteps are weighed equally, and the Min-SNR strategy [9]. As for the inverted cosine scheduler, we attempted uniform weighting and our proposed medium noise prioritization (MNP) weighting strategy, which downweighs losses from both low and high timesteps. We see that focusing on medium-level noise through the combined use of an inverted cosine scheduler and medium noise prioritization weighting strategy worked the best.

Feature Pooling Method: We investigated different methods to pool the two-dimensional output of the gait feature extractor into a one-dimensional condition—mean pooling, max pooling, and the sum of the mean and max. Out of the three pooling methods, mean pooling worked the best.

Inputs to Denoiser and Gait Feature Extractor: We investigated what would happen during diffusion pretraining

Table 6. Rank-1 accuracy on the Gait3D and GREW datasets for different values of r attempted during transfer learning. Values in bold denote the best results for each model on Gait3D and GREW.

Method	Gait3D			GREW		
	Rank-1 Accuracy (%)		Train Iter. ($\times 10^3$)	Rank-1 Accuracy (%)		Train Iter. ($\times 10^3$)
	✗ Data Aug.	✓ Data Aug.		✗ Data Aug.	✓ Data Aug.	
	$r = 0.1$			$r = 0.1$		
GaitGL	**34.4**	**34.4**	120 + 60	46.2	48.0	120 + 125
GaitPart	32.5	39.1	120 + 60	-	-	-
GaitSet	40.1	47.3	120 + 60	45.1	46.4	120 + 125
SMPLGait w/o 3D	49.1	57.2	120 + 60	43.8	44.5	120 + 125
GaitBase	47.5	55.8	120 + 60	50.7	52.7	120 + 90/120
	$r = 0.5$			$r = 0.5$		
GaitGL	33.1	32.3	120 + 60	54.3	56.5	120 + 125
GaitPart	**35.7**	**41.7**	120 + 60	-	-	-
GaitSet	**45.0**	**49.9**	120 + 60	50.9	53.3	120 + 125
SMPLGait w/o 3D	**53.4**	**60.7**	120 + 60	49.3	51.2	120 + 125
GaitBase	60.2	68.2	120 + 60	**58.5**	**62.0**	120 + 90/120
	$r = 1.0$			$r = 1.0$		
GaitGL	29.4	28.6	120 + 60	**56.3**	**58.6**	120 + 125
GaitPart	33.0	38.4	120 + 60	-	-	-
GaitSet	43.4	48.5	120 + 60	**52.0**	**55.4**	120 + 125
SMPLGait w/o 3D	51.1	58.6	120 + 60	**51.8**	**54.1**	120 + 125
GaitBase	**62.3**	**69.7**	120 + 60	58.5	61.6	120 + 90/120

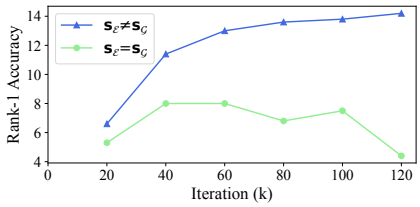

Fig. 5. Rank-1 accuracy curves during diffusion pretraining on Gait3D when $s_\mathcal{E} \neq s_\mathcal{G}$ and $s_\mathcal{E} = s_\mathcal{G}$ (SMPLGait w/o 3D).

should the input to the gait extractor $s_\mathcal{G}$ and the input to the autoencoder $s_\mathcal{E}$ be identical. Fig. 5 shows the rank-1 gait recognition accuracy curves of the gait feature extractor during the diffusion pretraining process when $s_\mathcal{E} \neq s_\mathcal{G}$ and $s_\mathcal{E} = s_\mathcal{G}$.

When $s_\mathcal{E} \neq s_\mathcal{G}$, we see a steady improvement in gait recognition accuracy during the diffusion pretraining. Yet, when $s_\mathcal{E} = s_\mathcal{G}$, the gait recognition accuracy fluctuates over time, suggesting that what the gait feature extractor is learning during diffusion pretraining is unlikely to be effective gait features. Rather, it could be extracting irrelevant reconstruction features to aid the denoiser in the denoising task.

Data Augmentation During Pretraining: We turned off data augmentation during diffusion pretraining to investigate the impact on downstream gait recognition performance. Without any data augmentation during pretraining, a significant drop in the downstream gait recognition performance is observed, highlighting the necessity of data augmentation during diffusion pretraining.

Size of Denoiser: We varied the size of the denoiser that is used during the diffusion pretraining process by changing its initial channel dimension. We see that the larger the denoiser, the better the downstream gait recognition performance. This

Table 7. Downstream rank-1 gait recognition accuracy on Gait3D for different diffusion pretraining hyperparameter settings. The first row shows the default setting we used for Gait3D. Highlighted entries denote the different hyperparameter settings compared to our default setting.

Diffusion Pretraining Hyperparameters							Gait3D
Noise Scheduler	Loss Weighting	Pooling	Data Augmentation	Denoiser Size	$l_{\mathcal{G}}/l_{\mathcal{D}}$	P_{uncond}	R-1 (%)
Inverted Cosine	MNP	Mean	✓	11.6 M	5	0.15	49.1
Inverted Cosine	Uniform	Mean	✓	11.6 M	5	0.15	47.6
Cosine	Uniform	Mean	✓	11.6 M	5	0.15	47.5
Cosine	Min-SNR	Mean	✓	11.6 M	5	0.15	48.2
Inverted Cosine	MNP	Max	✓	11.6 M	5	0.15	47.8
Inverted Cosine	MNP	Mean + Max	✓	11.6 M	5	0.15	47.4
Inverted Cosine	MNP	Mean	✗	11.6 M	5	0.15	44.6
Inverted Cosine	MNP	Mean	✓	3.7 M	5	0.15	47.5
Inverted Cosine	MNP	Mean	✓	40.2 M	5	0.15	**50.6**
Inverted Cosine	MNP	Mean	✓	11.6 M	1	0.15	45.9
Inverted Cosine	MNP	Mean	✓	11.6 M	2	0.15	47.4
Inverted Cosine	MNP	Mean	✓	11.6 M	5	0.00	47.2
Inverted Cosine	MNP	Mean	✓	11.6 M	5	0.50	47.7

suggests that further improvements can possibly be made to the downstream task by increasing the denoiser size during diffusion pretraining. That said, it would come at the cost of larger memory consumption and longer pretraining time.

Learning Rate of Denoiser and Gait Feature Extractor: We investigated how different relative ratios of the learning rate between the gait feature extractor and denoiser, $\frac{l_{\mathcal{G}}}{l_{\mathcal{D}}}$, affect the downstream performance. We kept the denoiser learning rate constant at 1e^{-4} and varied the learning rate of the gait extractor. We observe that increasing $\frac{l_{\mathcal{G}}}{l_{\mathcal{D}}}$ leads to better downstream performance.

Unconditional Training Probability: To investigate the effects of varying the unconditional training probability, P_{uncond}, during diffusion pretraining, we explored $P_{\text{uncond}} \in \{0, 0.15, 0.5\}$. We see that our proposed diffusion pretraining approach benefits from classifier-free guidance. However, too high an unconditional training probability ended up hurting the downstream gait recognition performance, likely as the gait feature extractor got updated less. A moderate unconditional training probability yielded the best result.

3. REFERENCES

[1] Chao Fan, Junhao Liang, Chuanfu Shen, Saihui Hou, Yongzhen Huang, and Shiqi Yu, "Opengait: Revisiting gait recognition towards better practicality," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2023, pp. 9707–9716.

[2] Beibei Lin, Shunli Zhang, Ming Wang, Lincheng Li, and Xin Yu, "Gaitgl: Learning discriminative global-local feature representations for gait recognition," *arXiv preprint arXiv:2208.01380*, 2022.

[3] Chao Fan, Yunjie Peng, Chunshui Cao, Xu Liu, Saihui Hou, Jiannan Chi, Yongzhen Huang, Qing Li, and Zhiqiang He, "Gaitpart: Temporal part-based model for gait recognition," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 14225–14233.

[4] Hanqing Chao, Yiwei He, Junping Zhang, and Jianfeng Feng, "Gaitset: Regarding gait as a set for cross-view gait recognition," in *Proceedings of the AAAI conference on artificial intelligence*, 2019, vol. 33, pp. 8126–8133.

[5] Jinkai Zheng, Xinchen Liu, Wu Liu, Lingxiao He, Chenggang Yan, and Tao Mei, "Gait recognition in the wild with dense 3d representations and a benchmark," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 20228–20237.

[6] Zheng Zhu, Xianda Guo, Tian Yang, Junjie Huang, Jiankang Deng, Guan Huang, Dalong Du, Jiwen Lu, and Jie Zhou, "Gait recognition in the wild: A benchmark," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 14789–14799.

[7] Alexander Quinn Nichol and Prafulla Dhariwal, "Improved denoising diffusion probabilistic models," in *International conference on machine learning*. PMLR, 2021, pp. 8162–8171.

[8] Drew A Hudson, Daniel Zoran, Mateusz Malinowski, Andrew K Lampinen, Andrew Jaegle, James L McClelland, Loic Matthey, Felix Hill, and Alexander Lerchner, "Soda: Bottleneck diffusion models for representation learning," *arXiv preprint arXiv:2311.17901*, 2023.

[9] Tiankai Hang, Shuyang Gu, Chen Li, Jianmin Bao, Dong Chen, Han Hu, Xin Geng, and Baining Guo, "Efficient diffusion training via min-snr weighting strategy," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 7441–7451.