Cross-view identification based on gait bioinformation using a dynamic densely connected spatial-temporal feature decoupling network

Cross-view identification based on gait bioinformation using a dynamic densely connected spatial-temporal feature decoupling network

Shuo Qiao[a], Chao Tang[a,*], Huosheng Hu[b], Wenjian Wang[c], Anyang Tong[d], Fang Ren[a]

[a] *School of Artificial Intelligence and Big Data Hefei University Hefei China*
[b] *School of Computer Science and Electronic Engineering, University of Essex, Colchester CO4 3SQ England, UK*
[c] *School of Computer and Information Technology, Shanxi University, Taiyuan, China*
[d] *School of Computer Science and Information Engineering, Hefei University of Technology, Hefei, China*

A b s t r a c t

Existing cross-view identification methods based on gait bioinformation often overlook the importance of feature reuse and the decoupling of spatial–temporal features in gait data. To address these challenges, we propose a novel approach named the Dynamic Densely connected Spatial-Temporal feature decoupling Network (DDSTFDN). First, the continuous gait sequence data are preprocessed by cropping and normalization before being fed into an initial network module to extract shallow gait features. These shallow features are then pro-cessed by the dynamic dense spatial–temporal decoupling network, which includes densely connected spa-tial–temporal feature decoupling blocks and enhanced convolutional block attention modules (E-CBAM) to obtain decoupled spatial–temporal features. Finally, the resultant gait features are divided into probe features and gallery features for similarity calculation, enabling accurate classification. Our approach achieves recogni-tion accuracies of 97.2 % and 87.6 % for the normal walking (NM) conditions of the CASIA-B and OUMVLP datasets, respectively, as well as 93.4 % and 78.1 % recognition accuracies in the walking with a backpack (BG) and walking with a coat or jacket (CL) walking complex scenarios in the CASIA-B dataset. In addition, our method obtains a recognition accuracy of up to 98.6 % on the CASIA-C dataset. On the CASIA-B dataset, we outperform the current baseline by 3.9 percentage points in accuracy for a batch size of 4×8, achieving a level of recognition comparable to that of the state-of-the-art (SOTA) approaches. The above experimental results demonstrate that DDSTFDN can effectively improve recognition accuracy and reduce resource consumption.

Keywords:
Identification; Gait bioinformation; Feature reuse; Spatial-Temporal decoupled features; Enhanced Convolutional Block Attention Module

1. Introduction

Biometrics identification is a technology for authentication by analyzing the physiological or behavioral characteristics of an individual, which includes gait recognition. Gait recognition aims to identify the human body and movements through its walking profile, stride, arm swing and other features. It has been successfully applied to many real-world applications [1,2]. As gait features can be easily collected than traditional biometric features, gait recognition methods have achieved impressive performance and have received widespread attention [1,3].

Current gait recognition methods can be divided into two categories.

- The template-based gait recognition method [4] that generates a gait template by compressing the information of a gait sequence from either original gait images or from gait feature maps. For instance, Chen et al. proposed a gait template named Chrono-Gait Image (CGI) [5], which employed a mapping function to synthesize a gait sequence into a single CGI image template. However, temporal information between gait sequences is ignored.
- The gait sequence-based recognition that uses gait sequence data for modeling [6]. For example, Chao et al. [7] regarded the gait walking process as a disordered sequence and used a 2D convolution- based method, namely GaitSet, to extract spatial information from the gait cycle to synthesize a gait feature template. However, the spatial–temporal information between gait sequences was overlooked.

Up to now, gait recognition remains a big challenging task in the real-world applications as its accuracy may be affected by many factors such as clothing, carrying items, cross-view, and complex scenes. Convolutional Neural Networks (CNNs) have been widely used in several fields due to their significant advantages in visual tasks [8–10]. In addition, Transformer [11–13] has recently demonstrated its strengths in long time-series modeling in vision tasks and has been applied to several vision tasks. In order to address the above issues, we have constructed a dynamic densely connected spatial–temporal feature decoupling network using CNNs and Transformer modules.

The main contributions of this paper can be summarized here.

(1) A Cross-view Identification based on Gait Bioinformation using a Dynamic Densely Connected Spatial-Temporal Feature Decoupling Network approach is proposed, which can directly convey shallow gait features to the deep network, enhancing the richness of features and improving the robustness of the model.
(2) A dynamic growth rate is deployed for the dynamic dense spatial–temporal decoupling feature extraction module, which can reduce the problem of gait feature information loss during feature extraction and enhance the generalization ability of the model.
(3) A dense spatial–temporal feature decoupling block is created to obtain spatial–temporal associated and decoupled features of gait. It utilizes a parallel structure to improve information expression capability so that low-rank convolution can more fully utilize spatial structural information and temporal correlation information.
(4) An enhanced convolutional block attention module is developed to focus on the detailed features of gait to for mining key features such as stride length, contour, and arm swing.
(5) A Residual-connected Transformer module (Res-T) is proposed, which can help the model to be modeled in terms of long time series while ensuring the convergence of the network, thus improving the representational power of the features.

2. Methodology

The proposed DDSTFDN encompasses four key stages: (i) initial data structure and preprocessing via the Initial Feature Processing Module (IFPM); (ii) dynamic dense spatial–temporal feature extraction in the Dense Spatiotemporal Feature Decoupling Block (DST Block) with an E-CBAM module; (iii) feature enhancement through the Feature Enhanced Processing Module (FEPM), incorporating mechanisms like Temporal Pooling (TP), Global Average Pooling and Global Max Pooling (GAP, GMP), Separate FC (SFC) layers, Batch Normalization Neck (BNNeck) and Res-T for improved representational power and generalization; (iv) loss function computation and label prediction. We will describe the above process in more detail below.

2.1. *IFPM*

2.1.1. *Data preprocessing*

a) Definition and Motivation: As shown in Fig. 1, the gait sequence in the initial feature processing module includes three processing procedures, which are size scaling, cropping transformation, and normalization. To reduce the computational load and eliminate irrelevant information, we extract regions of interest from the input video sequence that involve human motion (thus reducing network parameters and computational volume), and perform data normalization processing (improve the model's generalization performance).

b) Operation: Among a gait sequence $\boldsymbol{X} = \{\boldsymbol{X}_1, \boldsymbol{X}_2, ..., \boldsymbol{X}_T\}$, T is the number of sequence images contained in $\boldsymbol{X}$. Firstly, the gait sequence image $\boldsymbol{X}_i$ is resized, the image is then cropped to obtain the region of interest, and finally, the image data are normalized.

$$\boldsymbol{X} = \{N(C(R(\boldsymbol{X}_i)))|i = 1, 2, \ldots, T\} \tag{1}$$

where $\boldsymbol{X}_i$ a represents a frame image in the sample sequence, and $\boldsymbol{X} \in \mathbb{R}^{C\times T\times H\times W}$. $R(\bullet)$ represents the Resize function used for resizing the dimensions of the original sequence images. $C(\bullet)$ represents the Cut function used for cropping and transforming the images. $N(\bullet)$ represents the Normalize function used for normalizing the images.

Subsequently, training sets $D = \{\boldsymbol{X}_{in}^{(i)}, L_{in}^{(i)}|i = 1, 2, \ldots N\}$ is constructed, and $\boldsymbol{X}_{in}^{(i)} \in \mathbb{R}^{B\times C\times T\times H\times W}$. Where B represents the batch size, C represents the number of channels, T represents the number of frames, H represents the height of each frame image, W represents the width of each frame image, $\boldsymbol{X}_{in}^{(i)}$ represents the i-th sample set input to the network, and $L_{in}^{(i)}$ represents the label set of the i-th sample set.

2.1.1. *Initial block*

a) Definition and Motivation: The Initial Block in Fig. 1 contains two 3D convolutions, they can provide more meaningful information for follow-up networks. In addition, the time dimension and the number of network parameters are reduced, and resource consumption is decreased.

b) Operation: The inputs $\boldsymbol{X}_{in}^{(i)}$ are fed into an initial block where preliminary gait spatial–temporal feature extraction and temporal downscaling are performed to obtain shallow spatial–temporal features $\boldsymbol{X}_f$.

$$\boldsymbol{X}_f \leftarrow Ini(\boldsymbol{X}_{in}^{(i)}) \tag{2}$$

$$\boldsymbol{X}_f = C^{3\times1\times1}(C^{3\times3\times3}(\boldsymbol{X}_{in}^{(i)})) \tag{3}$$

where $Ini(\bullet)$ represents the calculation process of the Initial Block,

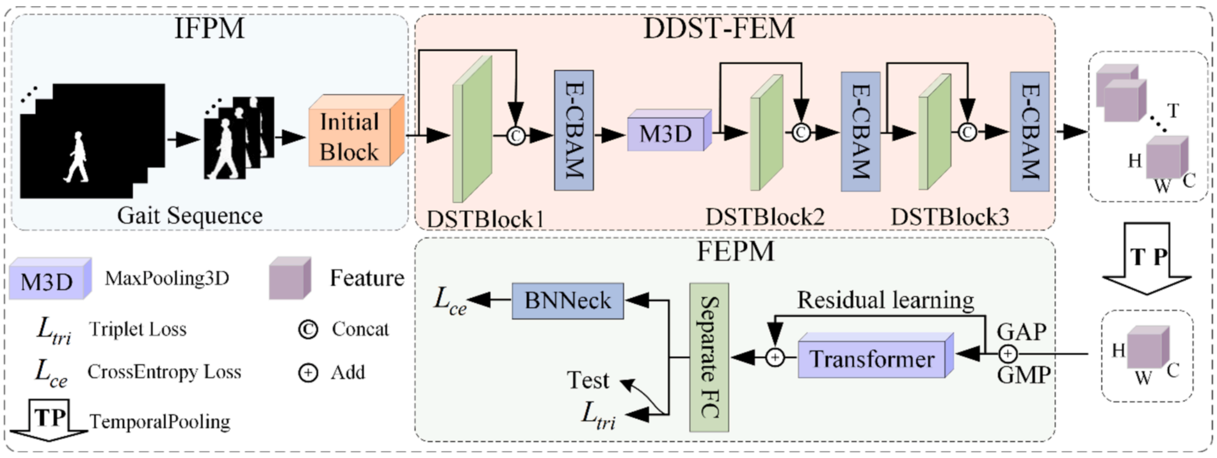


Fig. 1. The framework of the proposed DDSTFDN method.

$C(\bullet)$ stands for convolution operation and the corner markers are the dimensions of the convolution kernel, and $\boldsymbol{X}_f \in \mathbb{R}^{B\times C_0\times T_0\times H_0\times W_0}$.

2.2. *DDST-FEM*

DDST-FEM consists of three DST Blocks, three E-CBAMs, and a max pooling module. After the initial feature processing module, a shallow spatial–temporal feature $\boldsymbol{X}_f$ is obtained. $\boldsymbol{X}_f$ will be used as an input to the DDST-FEM and the output data y is obtained after 3 DST-A Blocks, 3 E-CABMs and 1 M3D processing (more details will be given below):

$$\boldsymbol{Y}_F \leftarrow DDST-FEM(\boldsymbol{X}_f) \tag{4}$$

Where $DDST-FEM(\bullet)$ represents the dynamic dense spatial–temporal decoupling feature extraction model and $\boldsymbol{Y}_F \in \mathbb{R}^{B\times C_4\times T_4\times H_4\times W_4}$.

2.2.1. *DST block*

a) Definition and Motivation: The DST Block consists of two branches. First, it uses the main feature extraction stream and the spatial–temporal decoupling feature extraction stream to extract distinctive features, respectively. Then, the outputs of the two branches are added together. Finally, the input is concatenated with the added output of the two branches to obtain the final output, as shown in Fig. 2(d).

We designed the DST block for two purposes. The first is to improve the recognition performance by combining shallow details in gait recognition (providing an advantage for the network to capture detailed features) with deep semantic features (providing an advantage for classification) through feature reuse. Secondly, since 3DConv may introduce irrelevant information with insufficient expressive power, we designed the DST Block to extract spatial–temporal relevant features (trunk) and spatial-temporally decoupled features (branches) using P3D [14].

b) Operation: As shown in Fig. 2, we designed two types of dense spatial–temporal feature extraction blocks. After experiments, we chose the DST-A Block as the dense spatiotemporal feature extraction block in the DDSTFDN method. Take DST Block1 as an example (Fig. 1), the input data is the output $\boldsymbol{X}_f$ of the initialization module.

(i) For the trunk feature extraction stream, the input data is $\boldsymbol{X}_f \in \mathbb{R}^{B\times C\times T\times H\times W}$, and $\boldsymbol{X}_f$ will pass through two traditional 3D convolution blocks (the trunk in Fig. 2(d)) in the trunk feature extraction stream to obtain the trunk output $\boldsymbol{Y}''_f$. The detailed calculation process can be expressed as:

$$\boldsymbol{Y}'_f = C^{1\times1\times1}(\boldsymbol{X}_f) \tag{5}$$

$$\boldsymbol{Y}''_f = C^{3\times3\times3}\left(\boldsymbol{Y}'_f\right) \tag{6}$$

where C^* represents a convolutional layer with a size of $*$, $\boldsymbol{Y}'_f \in \mathbb{R}^{B\times C_1\times T_1\times H_1\times W_1}$ and $\boldsymbol{Y}''_f \in \mathbb{R}^{B\times C_2\times T_2\times H_2\times W_2}$.

(ii) For the branch feature extraction stream, the input data is $\boldsymbol{Y}'_f$. $\boldsymbol{Y}'_f$ will go through two low-rank convolutions of the branch feature extraction stream (the branch in Fig. 2(d)), thereby obtaining the branch feature output $\boldsymbol{Y}'''_f$. The detailed calculation process is as follows:

$$\boldsymbol{Y}'_{f_1} = C^{1\times3\times3}\left(\boldsymbol{Y}'_f\right) \tag{7}$$

$$\boldsymbol{Y}'_{f_2} = C^{3\times1\times1}\left(\boldsymbol{Y}'_{f_1}\right) \tag{8}$$

$$\boldsymbol{Y}'_{f_3} = C^{1\times1\times1}\left(\boldsymbol{Y}'_{f_2}\right) \tag{9}$$

where $\boldsymbol{Y}'_{f_1} \in \mathbb{R}^{B\times C_1\times T_1\times H_1\times W_1}$, $\boldsymbol{Y}'_{f_2} \in \mathbb{R}^{B\times C_1\times T_1\times H_1\times W_1}$ and $\boldsymbol{Y}'_{f_3} \in \mathbb{R}^{B\times C_2\times T_2\times H_2\times W_2}$.

(iii) The outputs of the main trunk and branches are added to enhance the representation ability of features. Then, the input is concatenated with the added features to obtain the final feature representation $\boldsymbol{Y}_{F_1}$. The specific calculation process is as follows:

$$\boldsymbol{Y}_{F_1} \leftarrow Concat\left(\boldsymbol{Y}''_f \oplus \boldsymbol{Y}'_{f_3}, \boldsymbol{X}_f\right) \tag{10}$$

where $\oplus$ represents element-wise addition operation, $Concat(\bullet)$ represents concatenation operation, and $\boldsymbol{Y}_{F_1} \in \mathbb{R}^{B\times C_3\times T_3\times H_3\times W_3}$.

2.2.2. *E-CBAM*

a) Definition and Motivation: E-CBAM consists of an Enhanced Channel Attention Module (E-CAM) and a Spatial Attention Module (SAM). Gait recognition, as a task that requires attention to detailed features, suffers from the loss of gait feature information due to pooling operations. Therefore, we designed the E-CBAM module, which uses E-CAM to extract more expressive feature information and SAM to more effectively capture long-range dependencies and local information (as shown in Fig. 3), thereby obtaining more representative gait features.

b) Operation: The E-CBAM module includes two attention mechanisms, E-CAM and SAM, which process features in a serial manner. Taking the first E-CBAM module as an example for analysis, the input data is $\boldsymbol{Y}_{F_1} \in \mathbb{R}^{B\times C_3\times T_3\times H_3\times W_3}$. After passing through E-CBAM, the output is $\boldsymbol{Y}''_{F_1} \in \mathbb{R}^{B\times C_3\times T_3\times H_3\times W_3}$. The overall calculation formula is as follows:

$$\boldsymbol{Y}''_{F_1} \leftarrow E-CBAM(Y_{F_1}) \tag{11}$$

Detailed introductions for the two modules are provided separately. Firstly, E-CAM serves as the initial component for feature processing. The features obtained in the DST-A block, denoted as $\boldsymbol{Y}_{F_1}$, will be fed into

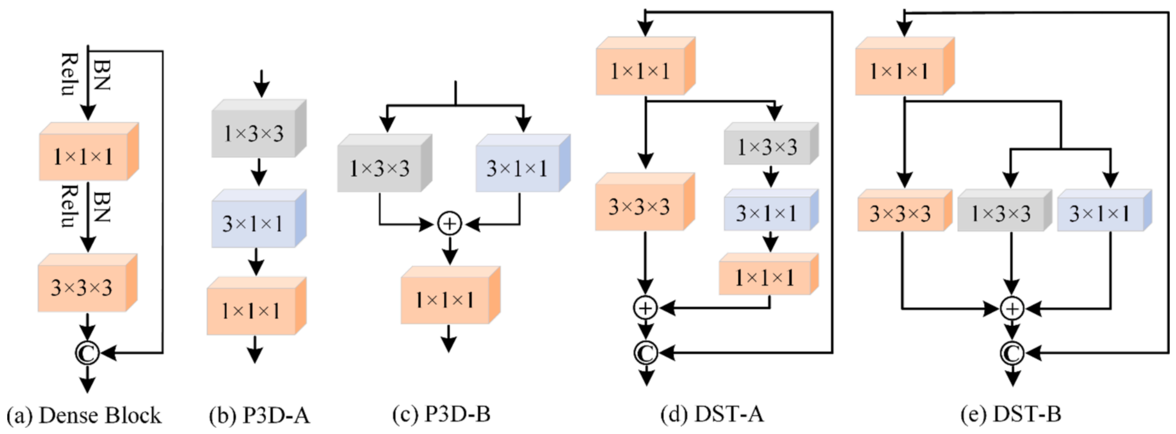
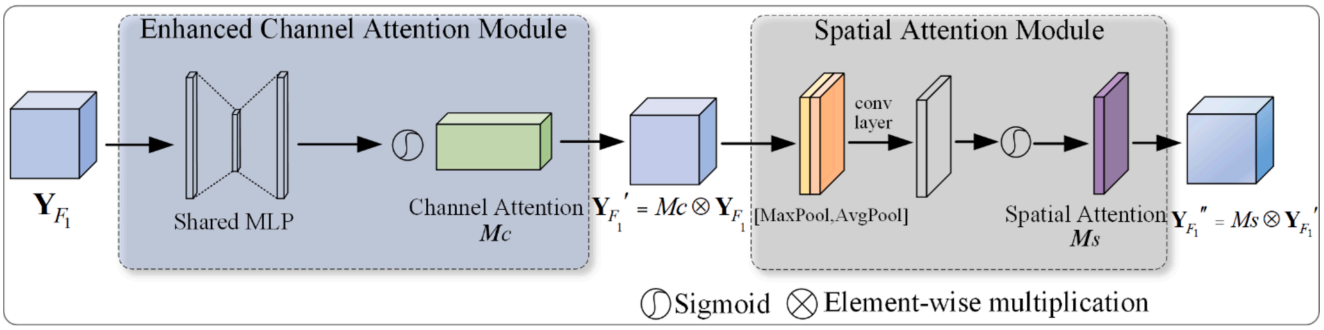


Fig. 2. Design overview of DST Block.



Enhanced Channel Attention Module
Spatial Attention Module
$\mathbf{Y}_{F_1}$
Shared MLP
Channel Attention Mc
$\mathbf{Y}_{F_1}' = Mc \otimes \mathbf{Y}_{F_1}$
conv layer
[MaxPool,AvgPool]
Spatial Attention Ms
$\mathbf{Y}_{F_1}'' = Ms \otimes \mathbf{Y}_{F_1}'$
Sigmoid
Element-wise multiplication

Fig. 3. Design overview of E-CBAM.

E-CAM, thereby setting different weights for the channel dimensions. The detailed calculation

process is as follows:

$$Sigmoid(x) = \frac{1}{1+e^{-x}} \tag{12}$$

$$Mc(\boldsymbol{Y}_{F_1}) = Sigmoid(MLP(\boldsymbol{Y}_{F_1})) \tag{13}$$

$$\boldsymbol{Y}_{F_1}' = Mc \otimes \boldsymbol{Y}_{F_1} \tag{14}$$

The *Sigmoid* function is used to assign weights to the data after passing through *MLP*, and then obtain the channel weight matrix *Mc* (formula 13). Then, the element-wise multiplication of *Mc* and feature $\boldsymbol{Y}_{F_1}$ is performed to obtain the final E-CAM module output $\boldsymbol{Y}_{F_1}' \in \mathbb{R}^{B\times C_3\times T_3\times H_3\times W_3}$.

Secondly, SAM will serve as another feature processing component following E-CAM. The output $\boldsymbol{Y}_{F_1}'$ of the E-CAM component will be used as input for the SAM module, thereby obtaining the spatial weight matrix *Ms*. The detailed calculation process is as follows:

$$f(x,y) = C^{7\times7\times7}(Concat(x,y)) \tag{15}$$

$$Ms\left(\boldsymbol{Y}_{F_1}'\right) = Sigmoid\left(f\left(MaxPool\left(\boldsymbol{Y}_{F_1}'\right), AvgPool\left(\boldsymbol{Y}_{F_1}'\right)\right)\right) \tag{16}$$

$$\boldsymbol{Y}_{F_1}'' = Ms \otimes \boldsymbol{Y}_{F_1}' \tag{17}$$

The *Sigmoid* function is used to assign weights to the features after passing through the function *f*, in order to obtain the spatial feature weight matrix *Ms* (formula 16). Finally, *Ms* is multiplied element-wise with the feature $\boldsymbol{Y}_{F_1}'$ to obtain the final output $\boldsymbol{Y}_{F_1}''$. In addition, the function *f* represents first performing a splicing operation, followed by a convolution operation with a kernel size of $(7\times7\times7)$ (formula 15), while *MaxPool* and *AvgPool* represent max pooling and average pooling, respectively.

2.3. *FEPM*

a) Definition and Motivation: FEPM includes TP, SFC, GAP, GMP, and BNNeck [15] (Fig. 4). We use FEPM as one of the modules of the method for the following two purposes: firstly, to reduce redundant information and enhance the representational ability of gait features; secondly, to enhance the model's representational power and generalization ability.

b) Operation: The detailed calculation process of FEPM is shown in Fig. 4, and the overall framework adopts a sequential and parallel structure. The input data for this module is $\boldsymbol{Y}_F$ (Formula 4). The overall calculation is as follows:

$$\boldsymbol{F}_{out} \leftarrow FEPM(\boldsymbol{Y}_F) \tag{18}$$

where FEPM refers to the entire feature processing module, $\boldsymbol{F}_{out}$ is the final output after passing through BNNeck, and $\boldsymbol{F}_{out} \in \mathbb{R}^{B\times C_5\times P}$,*P* represents the third dimension of the features obtained after the FEPM module, i.e., the horizontal feature dimension of $\boldsymbol{F}_{out}$.

After obtaining the feature out through FEPM, it will be processed through four modules, with the detailed calculation process as follows:

$$\boldsymbol{Y}_F' = TP(\boldsymbol{Y}_F) \tag{19}$$

$$\boldsymbol{Y}_F'' = \boldsymbol{Y}_F' \oplus \text{Transformer}(GAP(\boldsymbol{Y}_F') \oplus GMP(\boldsymbol{Y}_F')) \tag{20}$$

$$\boldsymbol{Y}_F''' = SFC(\boldsymbol{Y}_F'') \tag{21}$$

$$\boldsymbol{F}_{out} = BNNeck(\boldsymbol{Y}_F''') \tag{22}$$

where *TP* represents time dimension pooling, and the output is $\boldsymbol{Y}_F' \in \mathbb{R}^{B\times C_4\times H_4\times W_4}$. $\oplus$ represents element-wise addition operation, while *GAP* and *GMP* represent global average pooling and global maximum pooling, respectively, yielding the output $\boldsymbol{Y}_F'' \in \mathbb{R}^{B\times C_4\times P}$. *SFC* stands for separate fully connected, which is used for feature mapping operations, resulting in the output $\boldsymbol{Y}_F''' \in \mathbb{R}^{B\times C_4\times Q}$, *Q* is the horizontal feature dimension after *SFC*. The output $\boldsymbol{F}_{out}$ is obtained after passing through

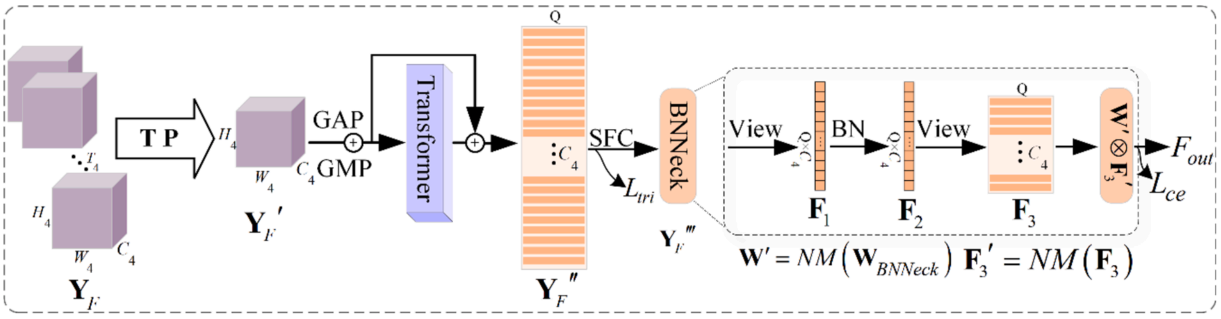


Fig. 4. Detailed display of Feature Enhancement Processing Module.

BNNeck.

The BNNeck component includes several key data processing procedures, which include dimension adjustment, batch normalization, and matrix dot product. The details of the dimensional changes are shown in Fig. 4. The detailed calculation process is as follows:

$$BN(\boldsymbol{X}) = BatchNorm(\boldsymbol{X}) \tag{23}$$

$$NM(\boldsymbol{X}) = Normalize(\boldsymbol{X}) \tag{24}$$

where *BN* represents a one-dimensional batch normalization operation, which is used to perform batch normalization processing on input features. *NM* is a normalization function, which is used to perform specified dimension normalization processing on data within the BNNeck component.

$$\boldsymbol{F}_1 = View\left(\boldsymbol{Y}_F'''\right) \tag{25}$$

$$\boldsymbol{F}_2 = BN(\boldsymbol{F}_1) \tag{26}$$

$$\boldsymbol{F}_3 = View(\boldsymbol{F}_2) \tag{27}$$

$$\boldsymbol{F}_{out} = NM(\boldsymbol{W}_{BNNeck}) \otimes NM(\boldsymbol{F}_3) \tag{28}$$

where $\boldsymbol{F}_1 \in \mathbb{R}^{B\times C'}$ and $C' = C_4 \times Q$. $\boldsymbol{F}_2 \in \mathbb{R}^{B\times C'}$ $\boldsymbol{F}_3 \in \mathbb{R}^{B\times C_4\times Q}$, $\boldsymbol{F}_{out}$, $\boldsymbol{W}_{BNNeck} \in \mathbb{R}^{B\times C_5\times Q}$. Note $\boldsymbol{W}_{BNNeck}$ is a constructed weighted matrix (used for changing data dimensions and enhancing feature representation), which requires parameter updating. The *View* function is used to adjust the data dimension.

2.4. Training and testing process

Training and Loss Function. For the training of the model, we adopt a general approach [6,7], sampling a batch size $P \times K$ from the training set each time, where P is the number of subjects sampled and K is the number of sequence samples per subject. In the model training process, for the output, we use triple loss function and cross entropy loss function to calculate the loss value of each horizontal feature component, respectively. Finally, the joint loss is used to optimize the model parameters by adding them together. The joint loss function can be expressed as:

$$Loss = L_{tri} + L_{ce} \tag{29}$$

We used a triple loss function in the model training phase[16]. In addition, each optimization sample consists of an anchor point (anc.), a positive sample (pos.) and a negative sample (neg.). Among them, the anchor point has the same label as the positive sample and different labels from the negative samples. To fully utilize the processed gait features, triplet loss calculation is performed on each horizontal feature component, and the calculation formula can be expressed as:

$$L_{tri} = \frac{1}{N_{tri}} \sum_{q=1}^{Q} \overbrace{\sum\nolimits_{i=1}^{P} \sum\nolimits_{a=1}^{K}}^{anc.} \overbrace{\sum\nolimits_{\substack{b=1 \\ b\neq a}}^{K}}^{pos.} \overbrace{\sum\nolimits_{\substack{j=1 \\ j\neq i}}^{P} \sum\nolimits_{n=1}^{K}}^{neg.} [m + dist]_+ \tag{30}$$

where N_{tri} is the number of all non-zero loss item triplets. Q is the scale of the horizontal component of each feature, which is also the Q-dimensional scale of F_{out}. m represents the margin parameter of the triplet loss function. *dist* represents the difference in distance from the anchor to the positive and negative examples in each triplet, which can be formulated as:

$$dist = D(F_{i,a}^q, F_{i,b}^q) - D(F_{i,a}^q, F_{j,n}^q) \tag{31}$$

where $F_{i,a}^q$ represents the q-th horizontal component feature in the a-th gait sequence sample of the i-th subject and is also the anchor point feature in the current triplet category. $F_{i,b}^q$ and $F_{j,n}^q$ have the same meaning as $F_{i,a}^q$, representing positive example features and negative example features, respectively.

Formula (31) represents the similarity measurement function, which is based on the Euclidean geometric distance and shown below:

$$D(\boldsymbol{A}, \boldsymbol{B}) = \|\boldsymbol{A} - \boldsymbol{B}\| \tag{32}$$

To better utilize the final gait horizontal feature components, we calculated the cross-entropy loss function value for each horizontal component feature. We used the average loss as the cross-entropy loss for each sample, and the specific calculation formula is as follows:

$$L_{ce} = -\frac{1}{N}\sum_{i=1}^{N}\frac{1}{Q}\sum_{q=1}^{Q}\sum_{k=1}^{K} y_{i,c,k}\log\left(p_{i,c,k}\right) \tag{33}$$

where N represents the number of samples in each batch, Q is the number of horizontal component features, K represents the number of categories, and $y_{i,c,k}$ is the symbol function (0 or 1), taking 1 if the

true category of sample i is the same as k, otherwise taking 0. $p_{i,c,k}$ is the predicted probability that the q-th horizontal component feature of the observed sample i belongs to category k.

Test. In the testing process, $\boldsymbol{Y}_F''' \in \mathbb{R}^{B\times C_4\times P}$ was used as the feature for calculating distances. Specifically speaking, the data of each subject will be divided into two sets: a gallery set and a probe set. The gallery set is denoted as G, and the probe set is denoted as P. The feature set of all samples in the image library is used as the area to be retrieved, and the samples in the probe set will be used to calculate the Rank-1 recognition accuracy. The above process can be formalized as:

$$Pred(p) = Label(g) \tag{34}$$

where $Pred(\bullet)$ represents the predicted label of the sample and $Label(\bullet)$ represents the given label of the sample.

For the similarity measure between p and the gallery sample, we use Euclidean distance to measure the similarity between the probe and the gallery features, which can be formulated as:

$$dist = \min\{D\left(p, G_g\right)\}, g = 1, 2, \ldots, G \tag{35}$$

where D is the Euclidean distance calculation function (formula (32)), p represents the current probe, G_g is the g-th gallery feature in the gallery set, and min represents taking the minimum value from the distance set.

3. Experiments

3.1. Datasets

CASIA-B [27] includes RGB and silhouette images of 124 individuals, totaling 13,680 sequence videos, and is a large-scale multi-view public gait dataset. The dataset considers three different walking conditions, i.e., NM, BG, and CL. The dataset was shot from 11 perspectives, covering a range of $0^\circ \sim 180^\circ$ (with one perspective every 18°). For fair comparisons, we follow the standard evaluation principles [4,6,7]. We use all data from 73 individuals (subject number 005 was removed due to missing data) for model training, while the remaining data from 50 individuals will be used to construct the gallery and probe sets for accuracy testing. For the testing phase, data with IDs NM#01–04 are used for generating the gallery set, while data with IDs NM#05–06, BG#01–02, and CL#01–02 are used for generating the probe set.

OUMVLP [28] is a multi-view large sample dataset, which is currently the largest public gait dataset, including 10,307 subjects. The shooting angles range from 0° to 90° (at intervals of 15°) and 180° to 270° (at intervals of 15°), totaling 14 perspectives, including two sequences (numbered #00–01). Due to its massive size, OUMVLP poses a significant challenge to our computational resources. Therefore, we followed the common dataset division ratio and selected a total of 512 subjects from the dataset. Among them, 256 subjects were used for training, and the remaining 256 subjects were used for testing. The data with the ID #00 was used for generating the gallery set, and the data with the ID #01 was used for generating the probe set.

CASIA-C is a large-scale database of images taken at night with infrared (thermal) cameras. The dataset contains infrared and silhouette images of 153 subjects and is a single-view (90°) public gait dataset. The dataset considers four different walking conditions, i.e., normal walking (fn), slow walking (fs), fast walking (fq), and walking with a bag (fb). In order to make a fair comparison, we use all the data from 100 individuals for model training, while the rest of the data from the remaining 53 individuals will be used to construct the gallery set and the probe set for accuracy testing. In the testing phase, the data numbered fn#01–02 were used to generate the gallery set and the data numbered fn#03–04, fs#01–02, fq#01–02 and fb#01–02 were used to generate the probe set.

3.2. Implementation details

We set the frame length of the sample to 30, the height of each frame image to 64, and the width to 44, with adopting data augmentation operations[24]. For the CASIA-B and CASIA-C dataset, we adopt data from 73 and 100 subjects as the training set, respectively. During training, the batch size is set to 32 ($P \times K = 4 \times 8$), and the maximum number of iterations is set to 80 K. The margin parameter of the triplet loss function is set to 0.2, and SGD is used as the optimize for the model. The initial learning rate and weight decay parameters of the optimizer are set to 0.1 and 0.0005, respectively. For the OUMVLP dataset, we selected data from 256 subjects as the training set, set the batch size to 32 ($P \times K = 8 \times 4$), and the maximum number of iterations to 120 K. The margin parameter of the triplet loss function is set to 0.2, and Adam is used as the optimize for the model. The initial learning rate is set to 1e-4. Regarding the configuration of the experimental environment and computer hardware, we are using the Windows 11 operating system, Python version 3.9, Torch version 2.0.1, an Intel i5-13600KF CPU, and an NVIDIA GeForce RTX 4080 GPU.

3.3. Experimental results

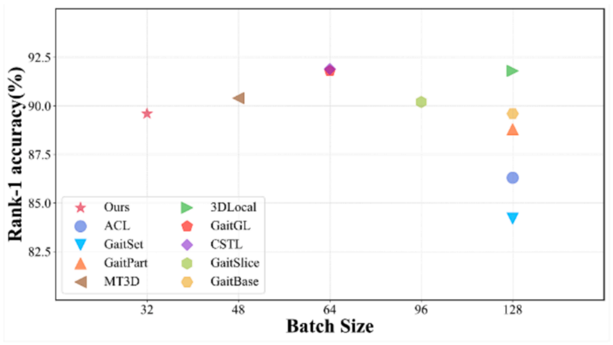
Experimental result on CASIA-B. In this section, the proposed DDSTFDN is compared with several state-of-the-art methods for recognition accuracy on the CASIA-B dataset. As can be seen from Table 1, our DDSTFDN has been compared with multiple methods, and its average performance surpasses most of the methods, reaching the current mainstream recognition accuracy level. Our method has obtained average recognition accuracies of 97.2 %, 93.4 %, and 78.3 % under the conditions of NM, BG, and CL, respectively. Compared with the methods listed in Table 1, our DDSTFDN method has achieved excellent recognition performance, which fully demonstrates its feasibility. From another perspective, our proposed method is analyzed and found to achieve excellent Rank-1 recognition accuracy at a smaller batch size. We have selected several state-of-the-art gait recognition methods for comparison, the methods involved include: ACL, GaitSet, GaitPart, MT3D, 3DLocal [8], GaitGL [32], CSTL [33], GaitSlice [34], and GaitBase. Fig. 5 shows that our method achieves exceptional recognition performance with small batch sizes. Under three walking conditions, our method achieved an average accuracy of 89.6 % with a batch size of 32 (Fig. 5(a)), and it reaches an average accuracy of 97.2 % under normal walking conditions alone (Fig. 5(b)). This demonstrates that our proposed methodology achieves excellent performance in a more environmentally friendly and low-carbon manner (reduced resource consumption).

Experimental result on OUMVLP. To validate the generality of our method, we have chosen the OUMVLP dataset as the generalization verification dataset. Table 2 shows the accuracy of our method in comparison with other methods, which shows our method reaching the level of current mainstream methods with a certain generalization ability.

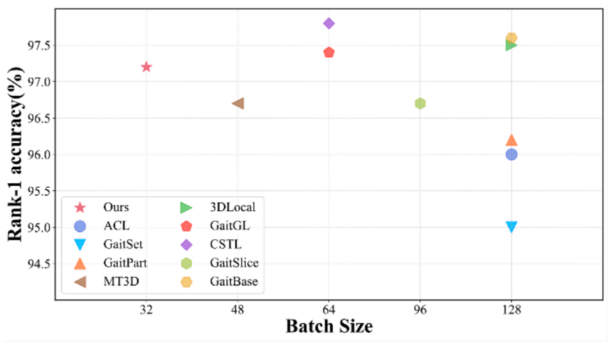
Experimental result on CASIA-C. To further validate the generality of our method, we also chose the CASIA-C dataset as another generalization validation dataset. Table 3 shows the accuracy of our method compared to other methods, and our method achieved 98.6 % accuracy. Using the experimental data table, our method achieves the highest recognition correctness rate so far on the CASIA-C dataset, which shows

Table 1
Cross-view average RANK-1 accuracies (%) on CASIA-B for different probe views excluding the identical-view cases.

Gallery NM#1–4			accuracy of view(0°∼180°)											Mean
Probe	Method	Source	0	18	36	54	72	90	108	126	144	162	180	
NM	GEINet[17]	ICB 2016	40.2	38.9	42.9	45.6	51.2	42.0	53.5	57.6	57.8	51.8	47.7	48.1
#5–6	CNN-LB[18]	TPAMI 2017	82.6	90.3	96.1	94.3	90.1	87.4	89.9	94.0	94.7	91.3	78.5	89.9
	GaitNet[19]	TPAMI 2022	93.1	92.6	90.8	92.4	87.6	95.1	94.2	95.8	92.6	90.4	90.2	92.3
	ACL[20]	TIP 2020	92.0	98.5	**100.0**	98.9	95.7	91.5	94.5	97.7	98.4	96.7	91.9	96.0
	GaitSet[21]	TPAMI 2022	91.1	99.0	99.9	97.8	95.1	94.5	96.1	98.3	99.2	98.1	88.0	96.1
	GaitPart[22]	CVPR 2020	94.1	98.6	99.3	98.5	94.0	92.3	95.9	98.4	99.2	97.8	90.4	96.2
	MvGAN[23]	TIP 2021	94.8	99.0	99.7	99.2	96.6	93.7	96.3	98.6	99.2	98.2	92.3	97.1
	MT3D[6]	MM 2020	**95.7**	98.2	99.0	97.5	95.1	93.9	96.1	98.6	99.2	98.2	92.0	96.7
	GaitBase[24]	CVPR 2023	94.8	**99.7**	99.8	**99.0**	96.8	95.3	97.3	99.2	**99.6**	**99.2**	**94.8**	**97.8**
	Ours	–	94.2	98.1	98.4	98.0	**96.9**	**96.3**	**99.5**	**99.3**	98.7	99.1	94.2	**97.2**
BG	GEINet[17]	ICB 2016	34.2	29.3	31.2	35.2	35.2	27.6	35.9	43.5	45.0	39.0	36.8	35.7
#1–2	CNN-LB[18]	TPAMI 2017	64.2	80.6	82.7	76.9	64.8	63.1	68.0	76.9	82.2	75.4	61.3	72.4
	GaitNet[19]	TPAMI 2022	88.8	88.7	88.7	94.3	85.4	**92.7**	91.1	92.6	84.9	84.4	86.7	88.9
	GaitSet[21]	TPAMI 2022	86.7	94.2	95.7	93.4	88.9	85.5	89.0	91.7	94.5	95.9	83.3	90.8
	GaitPart[22]	CVPR 2020	89.1	94.8	96.7	95.1	88.3	84.9	89.0	93.5	96.1	93.8	85.8	91.5
	MvGAN[23]	TIP 2021	92.4	94.7	97.2	94.6	88.7	83.6	87.8	93.8	96.3	95.2	86.8	91.9
	MT3D[6]	MM 2020	91.0	95.4	97.5	94.2	**92.3**	86.9	91.2	**95.6**	**97.3**	96.4	86.6	93.0
	GaitBase[24]	CVPR 2023	**93.6**	**96.4**	96.1	**95.6**	92.1	88.7	90.8	95.3	97.2	96.0	**90.7**	**93.9**
	Ours	–	91.6	95.1	**96.9**	94.2	92.0	89.2	**91.5**	94.5	**97.3**	**96.5**	88.8	**93.4**
CL	GEINet[17]	ICB 2016	19.9	20.3	22.5	23.5	26.7	21.3	27.4	28.2	24.2	22.5	21.6	23.5
#1–2	CNN-LB[18]	TPAMI 2017	37.7	57.2	66.6	61.1	55.2	54.6	55.2	59.1	58.9	48.8	39.4	54.0
	GaitNet[19]	TPAMI 2022	50.1	60.7	72.4	72.1	74.6	**78.4**	70.3	68.2	53.5	44.1	40.8	62.3
	GaitSet[21]	TPAMI 2022	61.4	75.4	80.7	77.3	72.1	70.1	71.5	73.5	73.5	68.4	50.0	70.4
	GaitPart[22]	CVPR 2020	**70.7**	**85.5**	**86.9**	**83.3**	77.1	72.5	76.9	**82.2**	83.8	80.2	66.5	**78.7**
	MvGAN[23]	TIP 2021	70.5	77.9	82.5	82.7	77.4	73.6	73.8	77.8	77.6	72.5	64.8	75.6
	GaitBase[24]	CVPR 2023	68.8	81.7	84.8	81.7	79.0	75.7	**78.0**	80.7	82.2	78.3	66.8	78.0
	Ours	–	70.1	83.4	84.6	81.2	**79.2**	74.2	76.0	81.0	**83.9**	**80.6**	**67.0**	**78.3**



(a) Average accuracy in the three walking conditions



(b) Average accuracy in the NM walking condition

Fig. 5. Comparison of batch size and Rank-1 accuracy on the CASIA-B dataset.

Table 2
Cross-view average RANK-1 accuracies (%) on OUMVLP excluding identical-view cases.

Probe(°)	Gallery All 14 views				
	GEINet[17]	DULE[25]	RPNet[26]	GaitSet[21]	**Ours**
0	11.4	56.2	73.5	**81.3**	80.2
15	29.1	73.7	84.4	88.6	**88.9**
30	41.5	81.4	89.6	90.2	**91.3**
45	45.5	82.0	89.8	90.7	**92.4**
60	39.5	78.4	86.3	88.6	**91.3**
75	41.8	78.0	87.4	89.1	**90.8**
90	38.9	76.5	86.0	88.3	**88.9**
180	14.9	60.2	76.3	**83.1**	82.7
195	33.1	72.0	83.2	**87.7**	85.6
210	43.2	79.8	88.6	**89.4**	89.0
225	45.6	80.2	88.9	**89.7**	87.9
240	39.4	76.7	85.7	**87.8**	87.5
255	40.5	76.3	86.4	**88.3**	86.0
270	36.3	73.9	84.4	**86.9**	84.3
Mean	35.8	74.7	85.0	**87.9**	**87.6**
Source	ICB 2016	CVPR 2021	TCSVT 2022	TPAMI 2022	–

that our method has excellent generalization ability.

3.4. Ablation studies

Our proposed method includes the setup of the network and several key components: DST Block, E-CBAM Block, BNNeck, Data Augmentation and Res-T. In this section, the effectiveness of these network settings and components are further investigated and analyzed.

The impact of network depth on experimental results. To investigate the effect of network depth on the recognition performance and the number of model parameters, we conducted two sets of controlled experiments: the Group (1) set used 3-layer DST blocks and E-CBAM blocks, and the Group (2) set used 4 layers. The results show that the deeper network (Fig. 6(a)) did not improve the performance; instead, the average recognition rate decreased by 0.8 % in the NM condition, 1.3 % in the BG condition, 2.8 % in the CL condition, and the Rank-1 recognition accuracy was also 1.6 % lower. Meanwhile, the number of model parameters (Fig. 6(b)) was significantly higher in the second group.

DST Block's impact on experimental results. In order to assess the impact of different blocks on the model performance, we conducted four sets of ablation experiments (Table 4): Group Base, Group (2), Group (3) and Group4. The results (Fig. 6(c)) show that the DST-A block performs best. The Rank-1 recognition accuracy of the DST-A block is 0.9 %, 1.7 %, and 4.5 % higher than that of the original 3D convolutional module and the DST-B block, and 1.2 %, 1.8 %, and 4.4 % higher than that of the original 3D convolutional module and the DST-B block, respectively, for the NM, BG, and CL conditions. Therefore, the two-branch design of the DST-A block is crucial for obtaining spatial–temporal correlation features and spatial–temporal decoupling features.

Attention mechanism's impact on experimental results. To assess the impact of different E-CBAM configurations, we conducted five sets of experiments (Group (1), Group (5), Group (6), Group (7), and Group (8) in Table 4). The results (Fig. 6(d)) show that the recognition accuracy of the model using E-CBAM is 0.5 %, 4.1 %, 11.6 %, and 0.9 % higher than the other groups, respectively. The attentional mechanism improves the model's ability to capture critical information and enhances robustness. Therefore, E-CBAM is an important component for improving recognition performance.

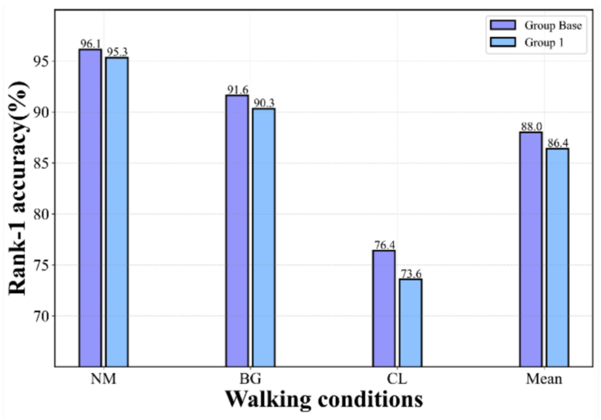
BNNeck's Impact on Experimental Results. To verify the impact of the BNNeck component, we conducted two sets of experiments. The results show that the average recognition rate of Group (9) using BNNeck is 0.1 %, 0.6 %, and 2.0 % higher than that of Group (1) without BNNeck in NM, BG, and CL conditions, respectively, and the Rank-1 accuracy is 1.0 % higher. Therefore, BNNeck is an important component for improving recognition performance.

The Impact of Data Augmentation and Res-T. In order to verify the effects of Data Augmentation and Res-T components on the model, we conducted three sets of experiments. The results show that the average Rank-1 recognition rate of the Group Best using Data Augmentation is 0.6 % higher than that of Group A without Data Augmentation. And the Rank-1 average recognition rate of the Group Best using Res-T is 1.0 % higher than that of Group B without Res-T. Thus, Data Augmentation and Res-T are important components for improving recognition performance.

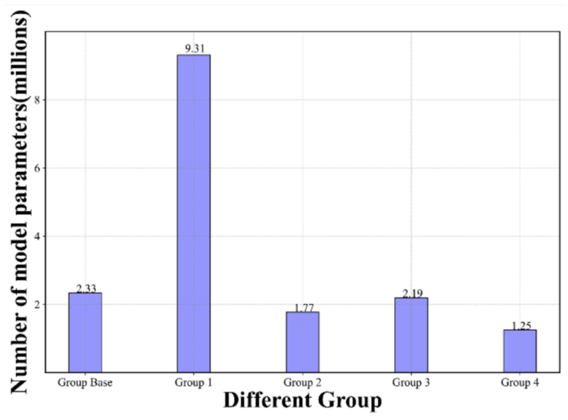
The Impact of different Batch Size. In order to investigate the effect of different batch size on experimental accuracy, we conducted

Table 3
Average RANK-1 accuracies (%) on CASIA-C.

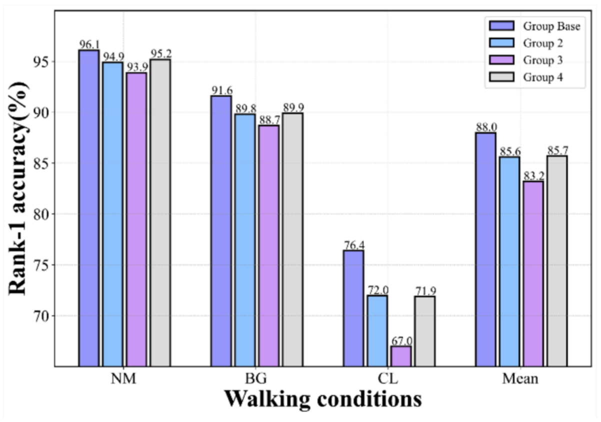
Gallery	Source	Year	Probe Accuracy				
			fn (#03–04)	fs (#01–02)	fq (#01–02)	fb (#01–02)	mean
fn(#01–02)	Hanif[29]	2023	93.6	91.5	95.8	**99.2**	95.0
	CTGait[30]	2024	–	–	–	–	97.7
	Anusha[31]	2024	**100.0**	**100.0**	96.0	94.0	97.5
	Ours	–	**100.0**	97.5	**100.0**	96.7	**98.6**



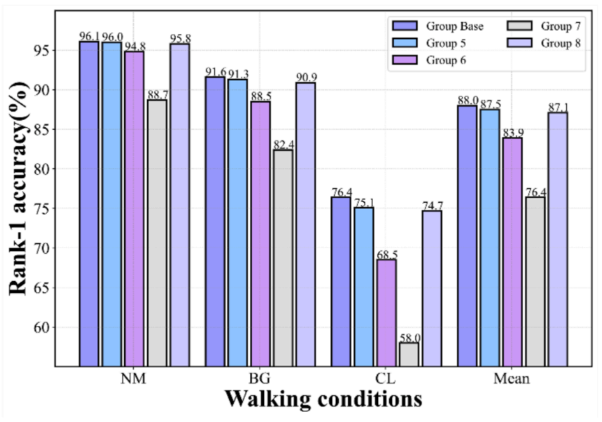
(a) Experimental effects of different model config



(b) Number of model parameters for different groups



(c) Experimental effects of different DST Block config



(d) Experimental effects of different Attention Block config

Fig. 6. Bar chart of the impact of model layer settings and model parameter amounts on experimental results (a and b). The impact of different DST Blocks and Attention Blocks on the model's recognition accuracy (c and d).

Table 4
Average Rank-1 recognition accuracy (%) for the DDSTFDN Ablation Experiment Cohort I on the CASIA-B dataset.

Group	Model Config		DST Block		E-CBAM	E-CAM	SAM	CBAM	BN Neck	Accuracy			
	(1,1,1)	(1,1,1,1)	DST-A	DST-B						NM	BG	CL	Mean
Base	√		√		√				√	**96.1**	**91.6**	**76.4**	**88.0**
1		√	√		√				√	95.3	90.3	73.6	86.4
2	√			√	√				√	94.9	89.8	72.0	85.6
3	√		√	√	√				√	93.9	88.7	67.0	83.2
4	√				√				√	95.2	89.9	71.9	85.7
5	√		√			√			√	96.0	91.3	75.1	87.5
6	√		√				√		√	94.8	88.5	68.5	83.9
7								√		88.7	82.4	58.0	76.4
8	√		√						√	95.8	90.9	74.7	87.1
9	√		√		√					96.0	91.0	74.4	87.1

three groups of experiments. The results show that the experimental accuracy is basically proportional to the batch size, and the accuracy of the experiments increases to different degrees as the batch size increases. According to the data in Table 5, the batch size of 4 × 8 (Group Best) is 0.7 % and 1.7 % more accurate than 4 × 6 (Group C) and 4 × 4 (Group D), respectively. Therefore, if the experimental conditions

Table 5
Average Rank-1 recognition accuracy (%) for the DDSTFDN Ablation Experiment Cohort II on the CASIA-B dataset.

Group	Data Augmentation	Res-T	Batch Size ($P \times K$)			Accuracy			
			4 × 4	4 × 6	4 × 8	NM	BG	CL	Mean
Best	√	√			√	**97.2**	**93.4**	**78.3**	**89.6**
A		√			√	96.8	92.3	77.9	89.0
B	√				√	96.6	92.0	77.3	88.6
C	√	√		√		96.7	92.5	77.6	88.9
D	√	√	√			96.1	91.6	75.9	87.9

allow, increasing the batch size appropriately will help to improve the performance of our model.

4. Conclusion

In this work, we propose a cross-view identification based on gait bioinformation using a dynamic densely connected spatial–temporal feature decoupling network approach. This method leverages the concept of feature reuse and introduces temporal and spatial low-rank convolutions to extract gait features in a decoupled manner. To further enhance the representation capability of gait features, an enhanced convolution block attention mechanism is incorporated, enabling the model to focus on more crucial features. Ablation experiments on benchmark datasets have verified that each component of our method contributes to the overall improvement in model performance, leading to the superior recognition capabilities of DDSTFDN.

Based on our findings, we hope that more researchers will focus on feature reuse. In addition to this, our work in this paper did not focus on the study of outdoor gait datasets, which is often the main scenario for applications in real-world application scenarios. Therefore, in our future work, we will further explore the outdoor gait dataset and complex scene generalization.

CRediT authorship contribution statement

Shuo Qiao: Writing – review & editing, Writing – original draft, Software, Methodology, Conceptualization. **Chao Tang:** Writing – review & editing, Supervision, Funding acquisition. **Huosheng Hu:** Writing – review & editing. **Wenjian Wang:** Writing – review & editing, Funding acquisition. **Anyang Tong:** Writing – review & editing, Funding acquisition. **Fang Ren:** Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This work was supported in part by the National Natural Science Foundation (62076154, U21A20513), the Provincial Graduate Academic Innovation Project (2022xscx145, 2023xscx145), the Innovative Research Program for Graduate Students of Hefei University (2024Ycxsj01, 2024Yxscx06), the Provincial Graduate Innovation and Entrepreneurship Practice Project (2023cxcysj191, 2023cxcysj192), and the Anhui Province Student Innovation Training Project (S202311059114, S202311059117, S202311059129, S202311059130, S202311059131).

Data availability

Data will be made available on request.

References

[1] C.S. Wan, L. Wang, V.V. Phoha, A Survey on Gait Recognition, ACM Computer Survey. 51 (5) (2019) 1–35, https://doi.org/10.1145/3230633.

[2] S. Zhang, J. Qi, et al., Fall-related gait pattern recognition based on surface electromyography using a hybrid neural network with transfer learning[J], Biomed. Signal Process. Control 98 (2024) 106771, https://doi.org/10.1016/j.bspc.2024.106771.

[3] A. Parashar, A. Parashar, W.P. Ding, et al., Deep learning pipelines for recognition of gait biometrics with covariates: a comprehensive review, Artif. Intell. Rev. 56 (8) (2023) 8889–8953, https://doi.org/10.1007/s10462-022-10365-4.

[4] Y.W. He, J.P. Zhang, H.M. Shan, et al., Multi-Task GANs for View-Specific Feature Learning in Gait Recognition, IEEE Trans on Information Forensics and Security. 14 (1) (2019) 102–113, https://doi.org/10.1109/TIFS.2018.2844819.

[5] W. Chen, Z. Junping, P. Jian, et al. Chrono-Gait Image: A Novel Temporal Template for Gait Recognition. Computer Vision, ECCV 2010: 11th European Conference, Heraklion, Crete, Greece, September 5-11, 2010, DAGM, IBM, NICTA, (2010) 257-270. Doi: 10.1007/978-3-642-15549-9_19.

[6] B.B. Lin, S.L. Zhang, et al., Gait Recognition with Multiple-Temporal-Scale 3D Convolutional Neural Network[C], in: 28th ACM International Conference on Multimedia (MM), 2020, pp. 3054–3062, 10.1145/3394171.3413861.

[7] H. Chao, Y. He, J. Zhang, et al., GaitSet: Regarding gait as a set for cross-view gait recognition, AAAI Conf. Artif. Intell. 33 (2019) 8126–8133.

[8] H. Zhen, X. Dixiu, S. Xu, et al., 3D Local Convolutional Neural Networks for Gait Recognition, In IEEE/CVF Confe-Rence on Computer Vision and Pattern Recognition Workshops (2021) 14900–14909, 10.1109/ICCV48922.2021.01465.

[9] Z. Zhu, S.-H. Wang, et al., A Survey of Convolutional Neural Network in Breast Cancer[J], CMES - Computer Modeling in Engineering and Sciences 136 (3) (2023) 2127–2172, https://doi.org/10.32604/cmes.2023.025484.

[10] B. Chen, C. Chen, et al., Computer Vision and Machine Learning-Based Gait Pattern Recognition for Flat Fall Prediction[J], Sensors 22 (20) (2022) 7960, https://doi.org/10.3390/s22207960.

[11] A. Vaswani, N. Shazeer, et al., Attention is all you need[C], in: Proceedings of the 31st International Conference on Neural Information Processing Systems, 2017, pp. 6000–6010.

[12] Z. Zhu, L. Liu, et al., OPT-CO: Optimizing pre-trained transformer models for efficient COVID-19 classification with stochastic configuration networks[J], Inf. Sci. 680 (2024) 121141, https://doi.org/10.1016/j.ins.2024.121141.

[13] S.-Y. Lu, Z. Zhu, et al., CTBViT: A novel ViT for tuberculosis classification with efficient block and randomized classifier[J], Biomed. Signal Process. Control 100 (2024) 106981, https://doi.org/10.1016/j.bspc.2024.106981.

[14] Q. Zhaofan, Y. Ting, M. Tao, Learning Spatio-Temporal Representation with Pseudo-3D Residual Networks, In IEEE/CVF Confe-Rence on Computer Vision and Pattern Recognition Workshops (2017) 5534–5542, https://doi.org/10.1109/ICCV.2017.590.

[15] Y. Sun, L. Zheng, Y. Yang, et al., Beyond part models: Person retrieval with refined part pooling (and a strong convolutional baseline), in: In European Conference on Comput-Er Vision (ECCV), 2018, pp. 480–496, 10.1007/978-3-030-01225-0_30.

[16] A. Hermans, L. Beyer, B. Leibe, In Defense of the Triplet Loss for Person Re-Identification, arXiv (2017), 1703.07737.

[17] K. Shiraga, Y. Makihara, D. Muramatsu, et al., Geinet: View-Invariant Gait Recognition Using a Convolutional Neural Network vol. 2016 (2016) 1–8, 10.1109/ICB.2016.7550060.

[18] Z.F. Wu, Y.Z. Huang, L. Wang, et al., A Comprehensive Study on Cross-View Gait Based Human Identification with Deep CNNs, IEEE Trans. Pattern Anal. Mach. Intell. 39 (2) (2017) 209–226, https://doi.org/10.1109/TPAMI.2016.2545669.

[19] Z.Y. Zhang, L. Tran, F. Liu, et al., On Learning Disentangled Representations for Gait Recognition, IEEE Trans. Pattern Anal. Mach. Intell. 44 (1) (2022) 345–360, https://doi.org/10.1109/TPAMI.2020.2998790.

[20] Y.Q. Zhang, Y.Z. Huang, S.Q. Yu, et al., Cross-View Gait Recognition by Discriminative Feature Learning, IEEE Trans. Image Process. 29 (2020) 1001–1015, 10.1109/10.1109/TIP.2019.2926208.

[21] H. Chao, K. Wang, Y. He, et al., GaitSet: Cross-View Gait Recognition Through Utilizing Gait As a Deep Set, IEEE Trans. Pattern Anal. Mach. Intell. 44 (7) (2022) 3467–3478, https://doi.org/10.1109/TPAMI.2021.3057879.

[22] F. Chao, P. Yunjie, C. Chunshui, et al., GaitPart: Temporal Part-Based Model for Gait Recognition, in: In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2020, pp. 14213–14221, 10.1109/CVPR42600.2020.01423.

[23] X. Chen, X.Z. Luo, J. Weng, et al., Multi-View Gait Image Generation for Cross-View Gait Recognition, IEEE Trans. on Image Processing. 30 (2021) 3041–3055, https://doi.org/10.1109/TIP.2021.3055936.

[24] C. Fan, J. Liang, C. Shen, et al., OpenGait: Revisiting Gait Recognition Toward Better Practicality, in: In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2023, pp. 9707–9716, 10.1109/CVPR52729.2023.00936.

[25] Zhang S, Wang Y, Li A. Cross-View Gait Recognition with Deep Universal Linear Embeddings. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 9091-9100. Doi: 10.1109/CVPR46437.2021.00898.

[26] Qin H, Chen Z, Guo Q, et al. RPNet: Gait Recognition With Relationships Between Each Body-Parts. IEEE Trans. on Circuits and Systems for Video Technology. 32 (5) (2022) 2990-3000. Doi: 10.1109/TCSVT.2021.3095290.

[27] Shiqi Y, Daoliang T, Tieniu T. A framework for evaluating the effect of view angle, clothing and carrying condition on gait recognition. In International Conference on Pattern Recognition, (2006) 431-444. https://doi.org/10.1109/ICPR.2006.67.

[28] N. Takemura, Y. Makihara, D. Muramatsu, et al., Multi-view large population gait dataset and its performance evaluation for cross-view gait recognition, IPSJ Trans. on Computer Vision and Applications. 10 (1) (2018), https://doi.org/10.1186/s41074-018-0039-6.

[29] C.-A. Hanif, M.-A. Mughal, et al., Human Gait Recognition Based on Sequential Deep Learning and Best Features Selection[J], Computers, Materials & Continua 75 (3) (2023) 5123–5140, https://doi.org/10.32604/cmc.2023.038120.

[30] Y. Liu, C.-Q. Wang, et al., Gait recognition of camouflaged people based on UAV infrared imaging[J], Infrared Phys. Technol. 138 (2024), https://doi.org/10.1016/j.infrared.2024.105262.

[31] Anusha R, Jaidhar CD. Speed-Invariant Gait Recognition Using Correlation Factor Lists for Classroom Attendance Systems[C]. 5th International Conference on Machine Learning, Image Processing, Network Security, and Data Sciences, MIND 2023, December 21, 2023 - December 22, 2023, 281-290 (2024).

[32] Lin B, Zhang S, Yu X. Gait Recognition via Effective Global-Local Feature Representation and Local Temporal Aggregation. In IEEE International Conference on Computer Vision, 14628-14636. Doi: 10.1109/ICCV48922.2021.01438.

[33] X. Huang, D. Zhu, H. Wang, et al., Context-Sensitive Temporal Feature Learning for Gait Recognition, In IEEE International Conference on Computer Vision (2021) 12889–12898, https://doi.org/10.1109/ICCV48922.2021.01267.

[34] H.K. Li, Y.D. Qiu, H.M. Zhao, et al., GaitSlice: A gait recognition model based on spatio-temporal slice features, Pattern Recogn. (2022) 124, https://doi.org/10.1016/j.patcog.2021.108453.