Multi-view Action Recognition

IT4851: HỆ CƠ SỞ DỮ LIỆU ĐA PHƯƠNG TIỆN

GIÁO VIÊN: NGUYỄN THỊ OANH

TRÌNH BÀY: LỮ MẠNH HÙNG

Nội dung trình bày

Giới thiệu

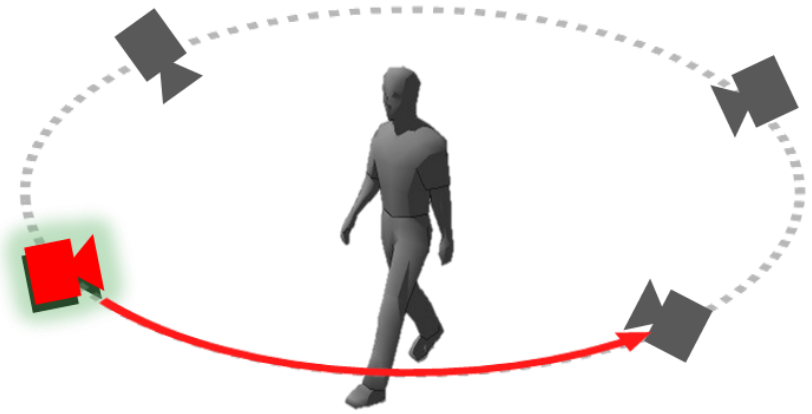
Các phương pháp

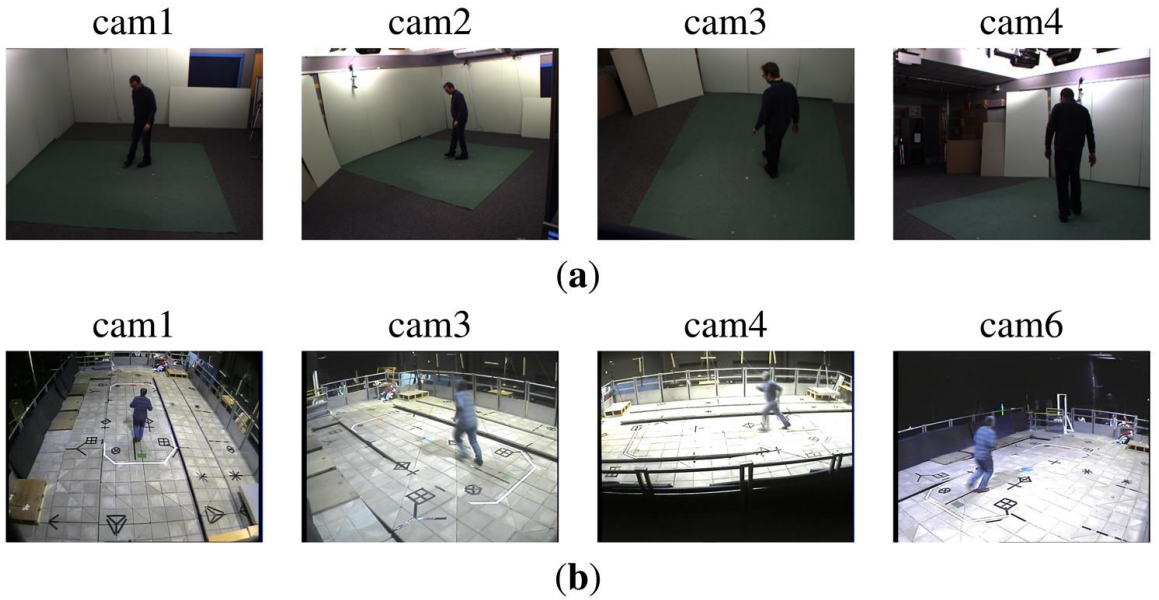
Kinh nghiệm

Mô hình đề xuất

Giới thiệu

- What?



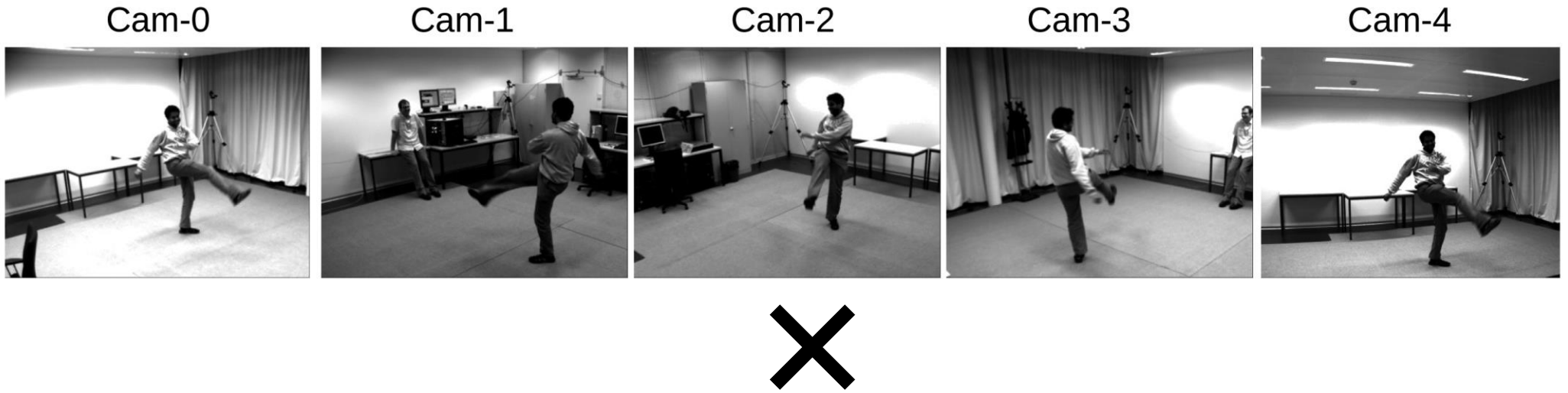


Giới thiệu

Why?

- Sự gia tăng của dữ liệu multi-view
- Đòi hỏi sự chính xác cao khi nhận diện hành động

➢ Ứng dụng trong các hệ thống giám sát, an ninh



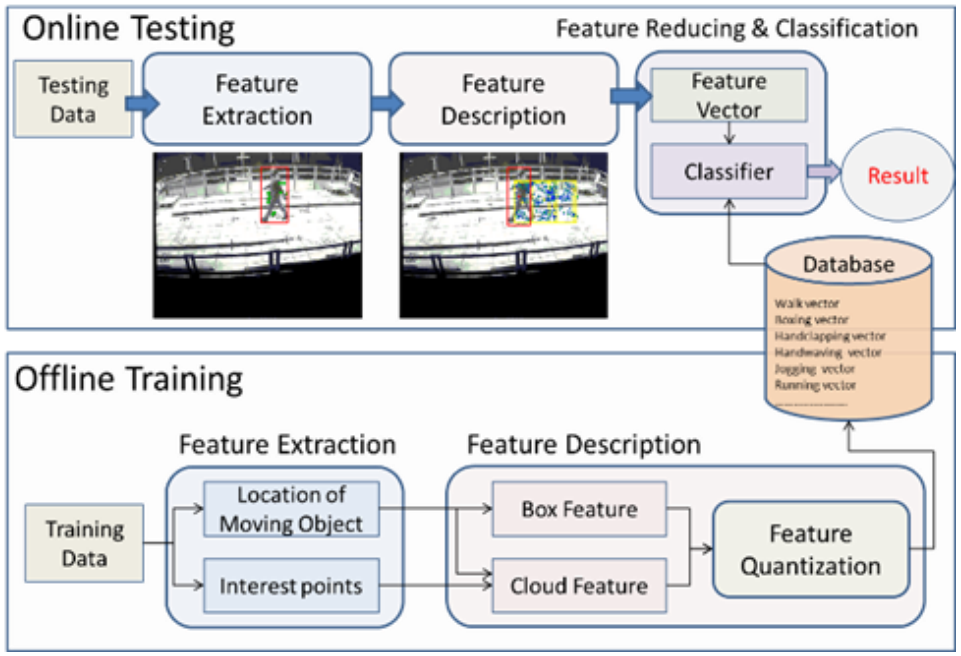
Giới thiệu

How?

- Hệ thống cần hoạt động hiệu quả khi góc quay thay đổi

Phân loại các phương pháp	Kỹ thuật xử lý ảnh cổ điển	Deep neural networks
Kết hợp các view		✓
Trích được các đặc trưng bất biến với view	✓	✓

Feature-based Automated Multi-view Human Action: Box feature + Cloud feature



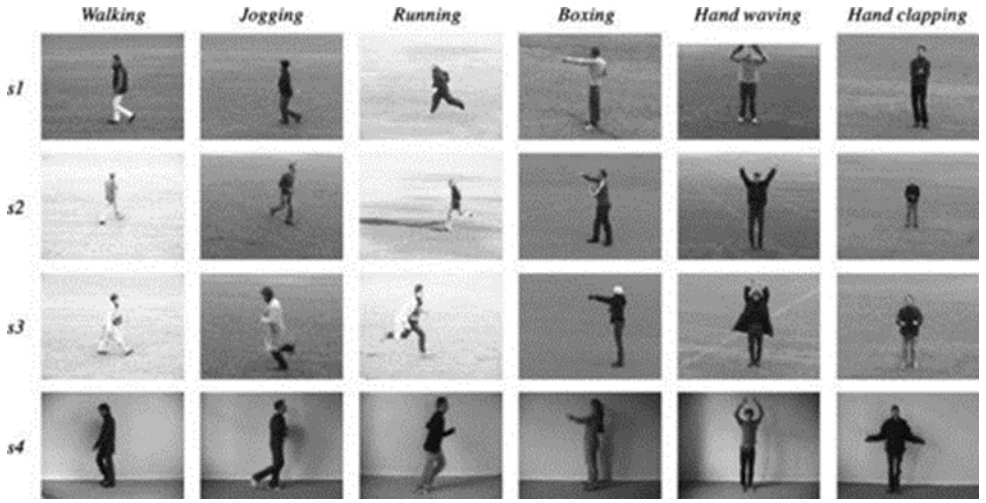
System Overview

Phương pháp nắm bắt thông tin local và global temporal để đưa ra sự phân bố của interesting points

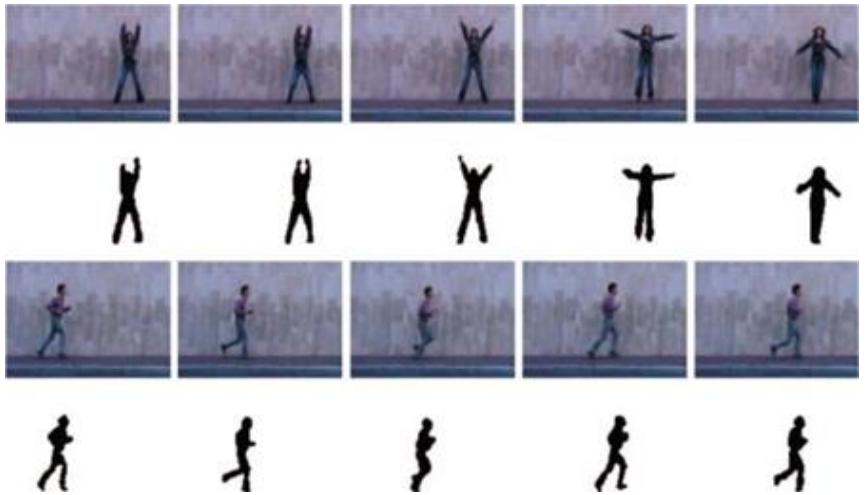
- Gán nhãn thời điểm bắt đầu và kết thúc chuỗi action một cách tự động
- Sử dụng interesting points để xác định các motion nên chi phí tính toán thấp
- Áp dụng view-invariant feature để định vị action trong multi-view do đó không bị ảnh hưởng khi view thay đổi

Datasets

KTH: 6 ACTION CLASSES X 4 SCENARIOS X 25 VIDEOS = 600 VIDEOS

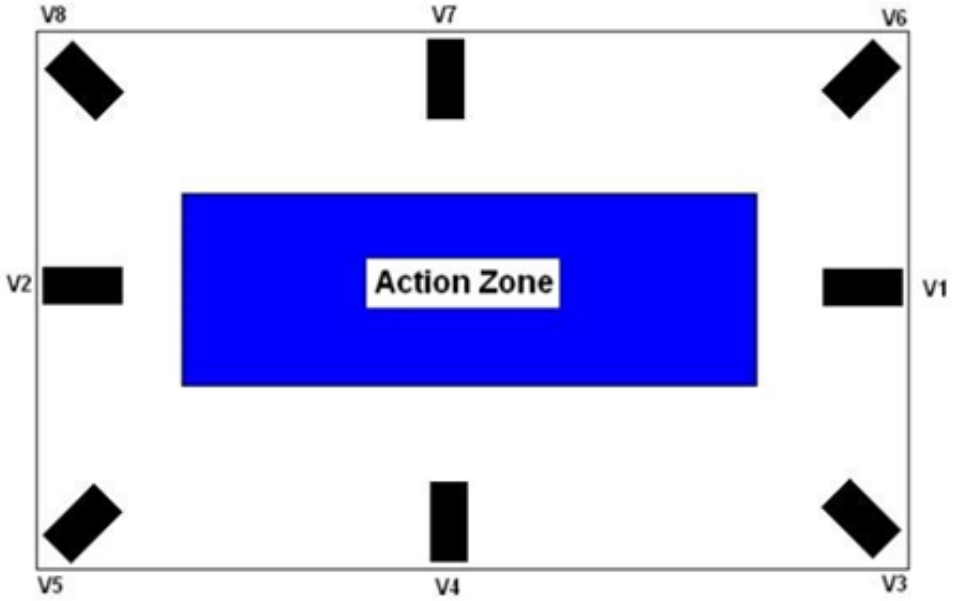


WEIZMANN: 90 VIDEOS (9 SUBJECTS, 10 ACTIONS)



Datasets

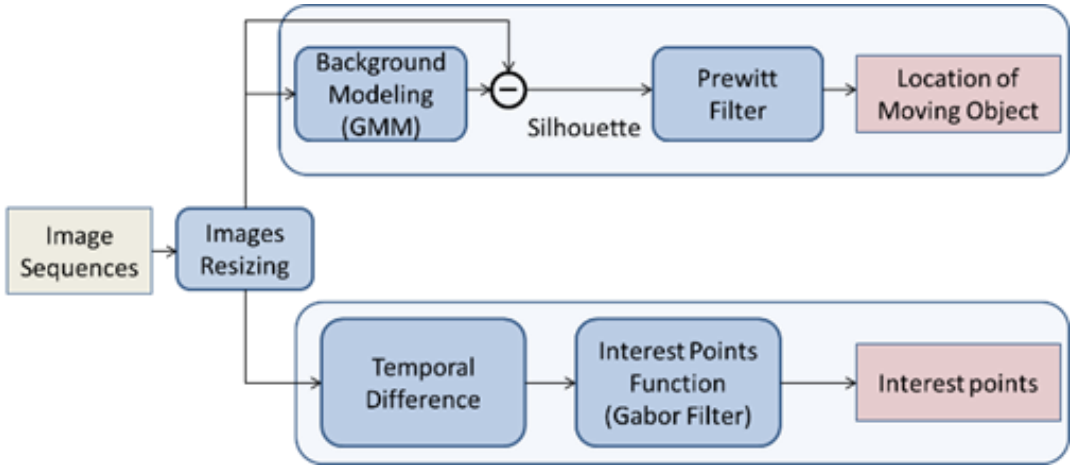
MUHAVI : MULTI-VIEW VIDEOS VỚI 14 PEOPLE X 17 ACTIONS X 8 VIEWS



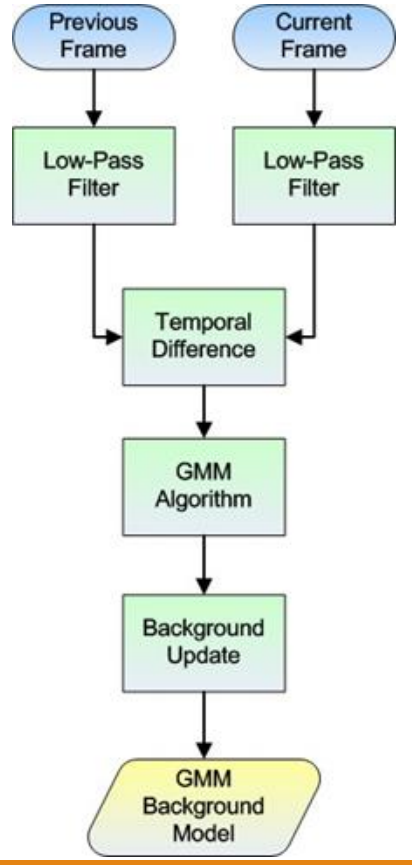
Feature extraction

Moving object localization

Overview of feature extraction



GMM background model construction



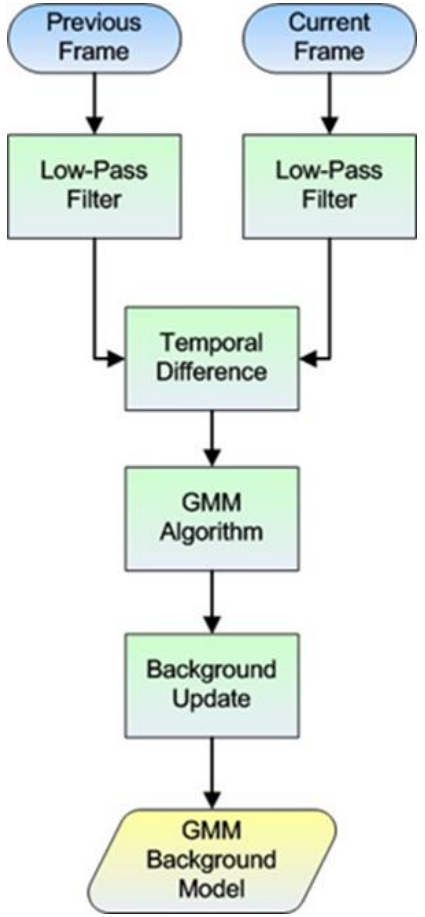
Feature extraction

Moving object localization

Đầu tiên, sử dụng bộ lọc low-pass để giảm nhiễu

Mô hình hóa cường độ của mỗi pixel bằng phương pháp GMM với K Gaussian distributions

(Với K lớn hiệu quả mô hình hóa với GMM sẽ tốt hơn)



Feature extraction

Moving object localization

Xác xuất một pixel có giá trị cường độ X_t:

$$P(X_t) = \sum_{k=1}^{K} \omega_{k,t} \cdot \eta(X_t, \mu_{k,t}, \sum_{k,t})$$

K: số lượng distributions

$\omega_{k,t}$: trọng số của *k*-th Gaussian

Hàm phân phối: $\eta(X_t, \mu_t, \sum_t) = \frac{1}{(2\pi)^{n/2} |\sum_t|^{1/2}} \exp\{-\frac{1}{2}(X_t - \mu_t)^T \sum_t^{-1} (X_t - \mu_t)\}$

Ma trận hiệp phương sai (tính đơn giản): $\sum_{k,t} = \sigma_k^2 \mathrm{I}$

Feature extraction

Moving object localization

Sử dụng *Temporal diference* để trích xuất vùng background và update các pixel trong vùng

Sắp xếp các Gaussian distributions theo giá trị ω/σ

Chọn ra B distributions đầu tiên để làm background model

$$B = \arg\min_{b}\left(\sum_{k=1}^{b}\omega_{k,t} > T\right)$$

Feature extraction

Moving object localization

Khi một pixel mới được thêm vào (có cường độ X_{t+1}) sẽ được kiểm tra lần lượt K distribution.

Nếu giá trị xác xuất nằm trong khoảng độ lệch chuẩn thì pixel đó được coi là nền.

Sau đó cập nhật lại giá trị:

$$\omega_{k,t+1} = (1-\alpha)\omega_{k,t} + \alpha(M_{k,t+1})$$

$$\mu_{t+1} = (1-\rho)\mu_t + \rho X_{t+1}$$

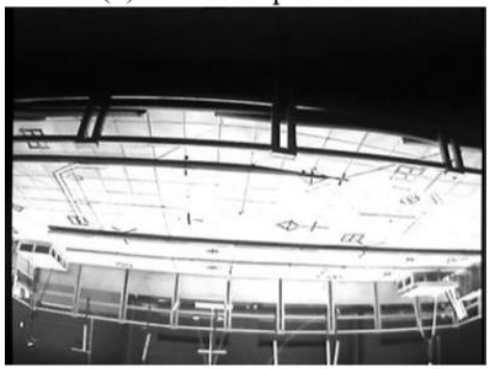
$$\sigma_{t+1}^2 = (1-\rho)\sigma_t^2 + \rho(X_{t+1}-\mu_{t+1})^T(X_{t+1}-\mu_{t+1})$$

$$\rho = \alpha\eta(X_{t+1} \mid \mu_{k,t}, \sigma_{k,t})$$

Feature extraction

- Moving object localization

(a) Video Sequence



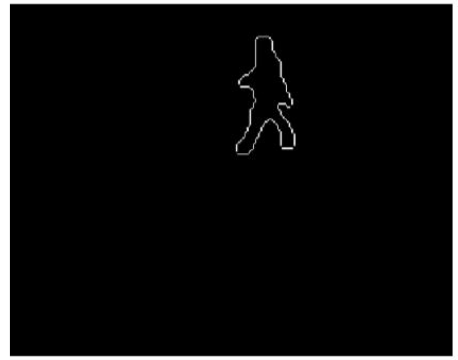
(b) GMM Background Image

Background image construction by GMM

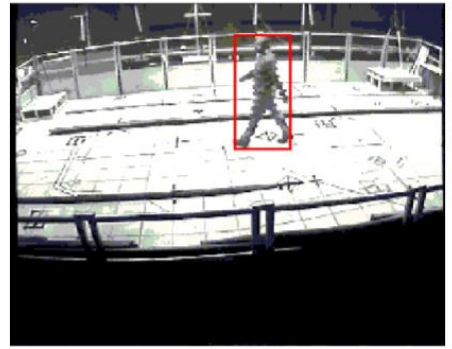
(a) Current Image

(b) Silhouette

Silhouette obtained by background subtraction



(a) Location of Moving object



(b) Bounding Box

Fig. 10: Moving object obtained by Prewitt filter

Feature extraction

Extraction of Points of Interest

Mục tiêu: khoanh vùng action

VD: Bounding box vùng tay đối với action “boxing”

Bregonzio et al. đề xuất:

- *Frame different* đối với regions of interest
- *2D Gabor filters* được sử dụng để lọc trên regions of interest

➢ Tách được các đặc trưng về không gian và thời gian

Feature extraction

Extraction of Points of Interest

2D Gabor filters

2 thành phần: Gaussian kernel function modulated by a *sinusoidal plane wav*

$$g(x,y) = s(x,y)\,G(x,y)$$

- Complex sinusoid (carrier): $s(x,y) = cos[2\pi(\mu_0 x + v_0 y) + \theta_i]$
- Gaussian-shaped function (envelope): $G(x,y) = exp(\frac{x^2+y^2}{2\rho^2})$

  $G(x,y)$ được điều chỉnh bởi tham số p: $\mu_0 = v_0 = \frac{1}{2\rho}$

➢ p là tham số duy nhất điều chỉnh khả năng scale của bộ lọc (11 pixels)

Feature extraction

Extraction of Points of Interest

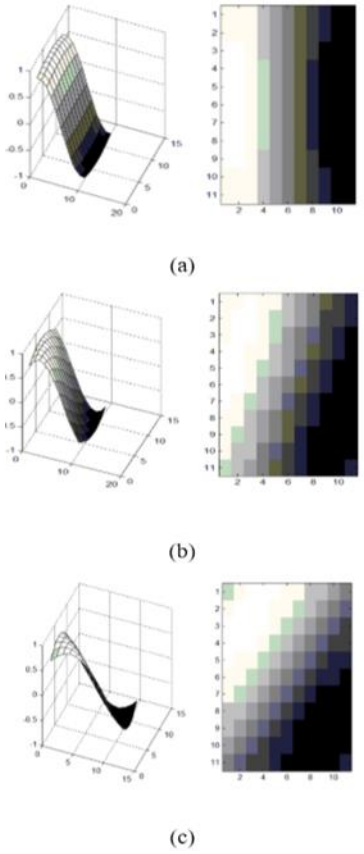
2D Gabor filters $g(x, y)$ * bounding boxes

Tạo một ngưỡng để chọn ra các points of interest

- Biểu diễn được các đặc trưng của action bằng points of interest

Feature extraction

Extraction of Points of Interest



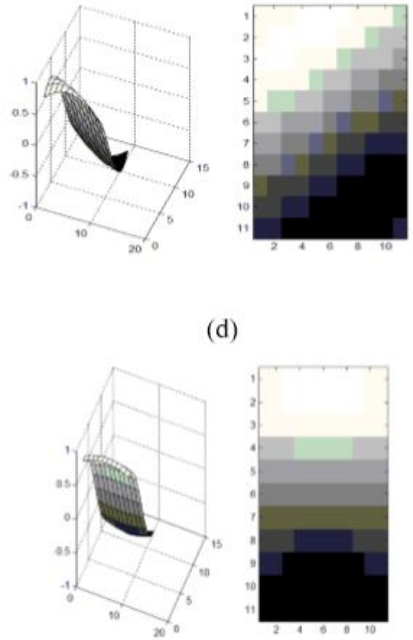


Fig. 11: Examples of the 2D Gabor filters oriented along (a) 0°, (b) 22:5°, (c) 45°, d) 67:5° and (e) 90°

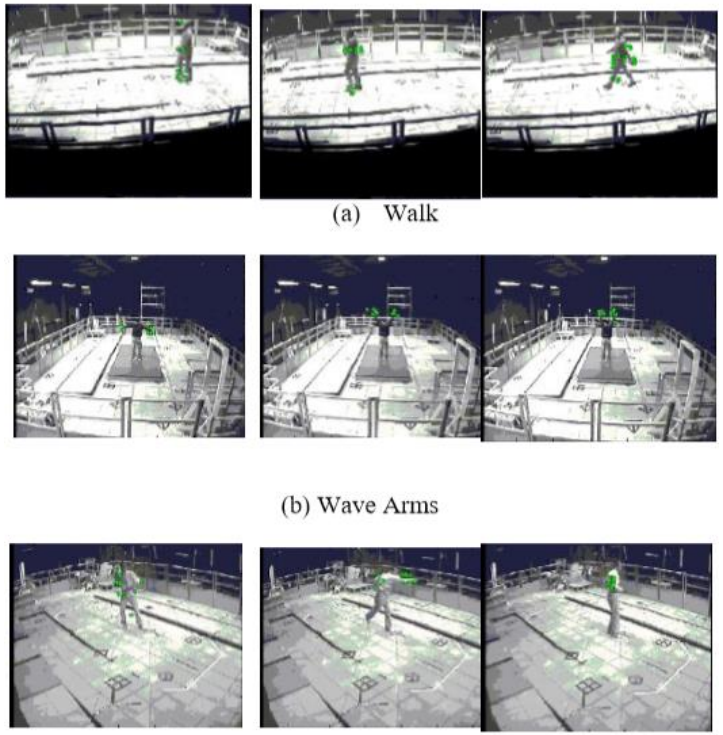


Fig. 12: Results of interest point detection

Feature extraction

Feature description

Nội dung:

Biểu diễn feature vectors

Mô tả cách giảm chiều không gian của các feature vector

A. Box feature

Global feature thể hiện hình dáng và tốc độ của foreground object

Feature B_t^r: tỉ lệ chiều cao và chiều rộng của object

Feature B_t^{Sp}: tốc độ tuyệt đối được chuẩn hóa bởi chiều cao object

Feature extraction

Feature description

B. Cloud feature

Point of interest chứa spatial information

Point of interest được tách từ các frame liên tiếp nên có temporal information

➢ Tập hợp các point of interest tạo thành point cloud chứa spatial và temporal information

Feature extraction

Feature description

B. Cloud feature

Video A gồm T frame: $A = [I_1, \ldots, I_t, \ldots, I_T]$

Ns: kích thước temporal scale

K: kích thước cumulative scale tích lũy $[I_{t-NS}, \ldots, I_t], [I_{t-2\times NS}, \ldots I_t],$ $\ldots,[I_{t-K\times NS}, \ldots, I_t]$

Cloud point của frame I_t: $[C^1, \ldots, C^S, \ldots, C^K]$

Feature extraction

Feature description

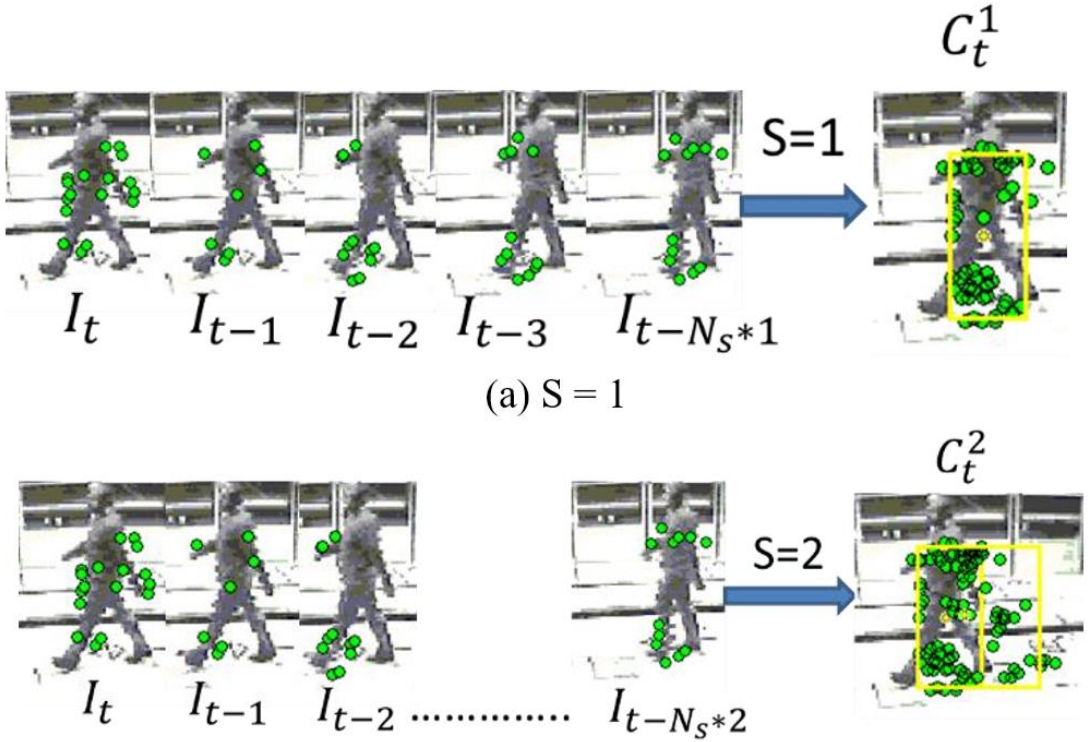


Fig. 13: Cloud for different temporal scale S

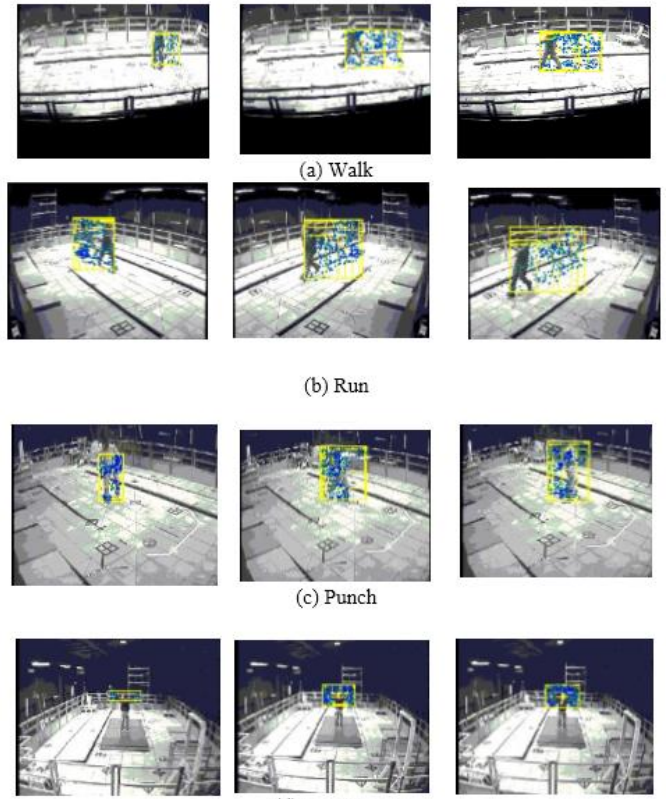


Fig. 14: Examples of the interest points clouds

Feature extraction

Feature description

B. Clound feature

Biểu diễn của s-th scale cloud: $[C_s^r, C_s^{Sp}, C_s^{Vd}, C_s^{Hd}, C_s^{Hr}, C_s^{Wr}]$

C_s^r: tỉ lệ chiều cao và rộng của cloud

C_s^{Sp}: tốc độ tuyệt đối của cloud

C_s^{Vd}: khoảng cách dọc giữa tâm object và cloud

C_s^{Hd}: khoảng cách ngang giữa tâm object và cloud

C_s^{Hr}: tỉ lệ giữa chiều cao của object và cloud tương ứng

C_s^{Wr}: tỉ lệ giữa chiều rộng của object và cloud tương ứng

Feature extraction

Feature description

Kết quả: Mỗi frame có 6S + 2 features

S: số lượng scale

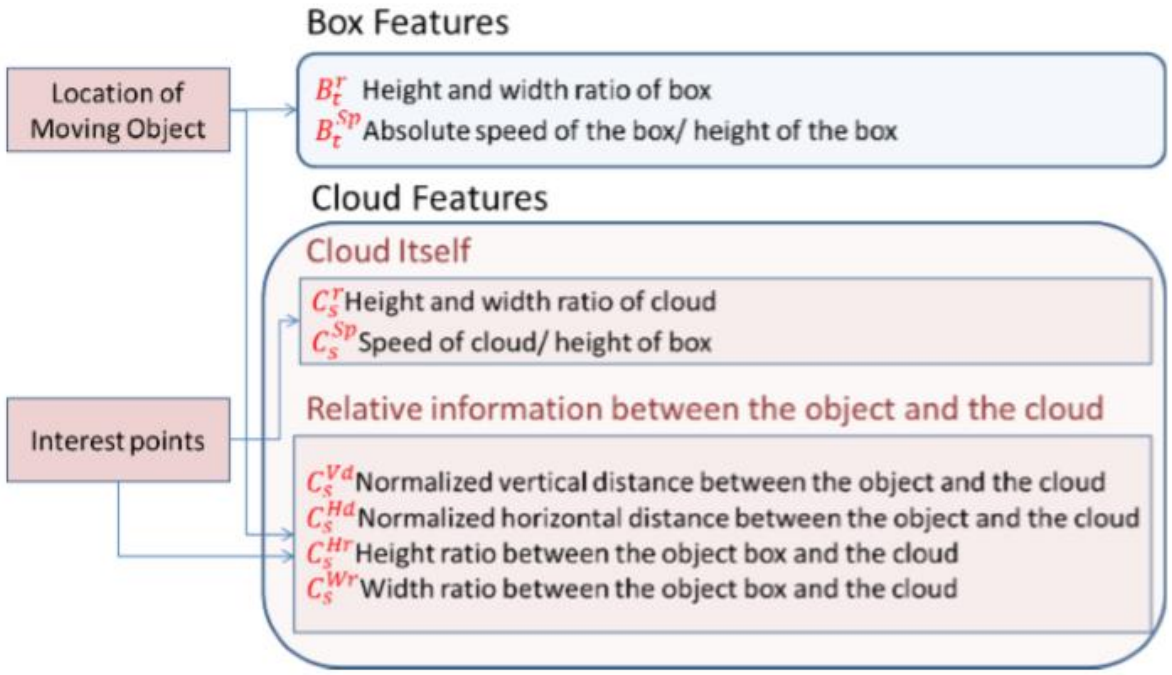


Fig. 15: Overview of the features of the proposed approach

Feature extraction

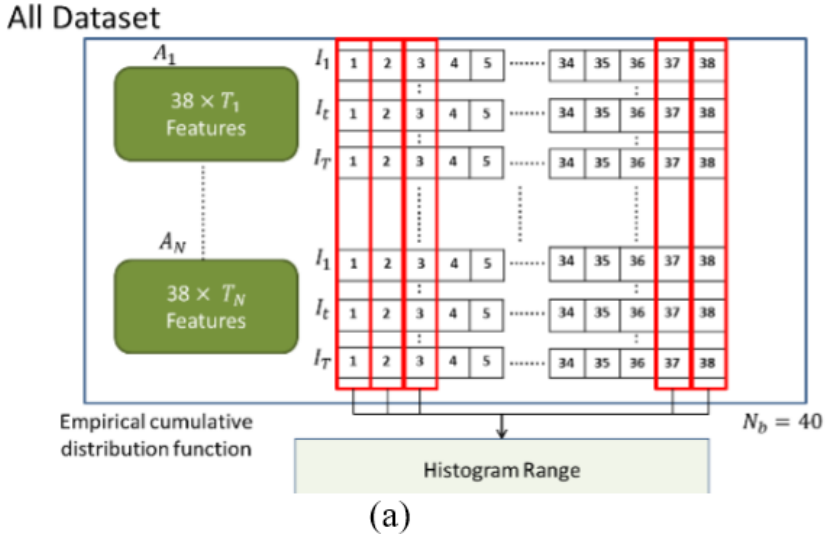
Feature description

C. Quantization

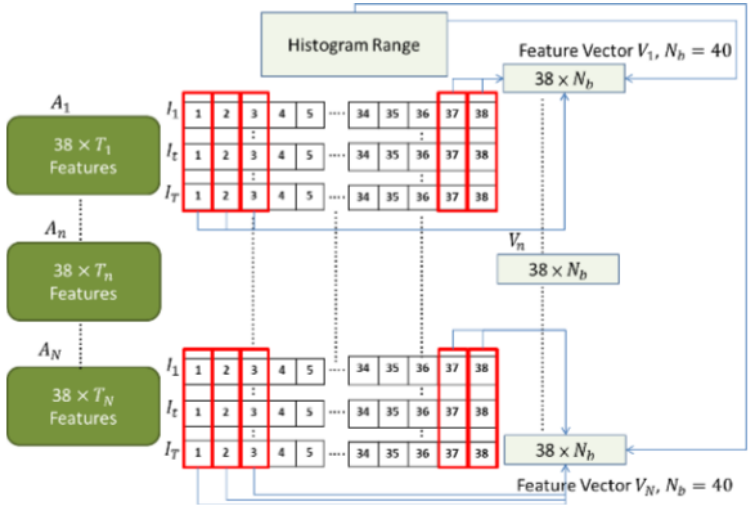
Chuỗi action: (6S + 2)T features => kích thước không gian feature lớn

Gây ra: over fitting, giảm hiệu năng

- Giảm kích thước không gian feature



(a)



(b)

Fig. 16: Overview of quantization (a) A histogram range is produced by observing one of all features in all the dataset separately. (b) Each action sequence A is represented as $(6S+2)$ N_b features.

Feature reduction and classification

Nearest Neighbor Classifier

K = 5 - WEIZMANN dataset

K = 3 - KTH dataset

K = 6 - MuHAVi dataset

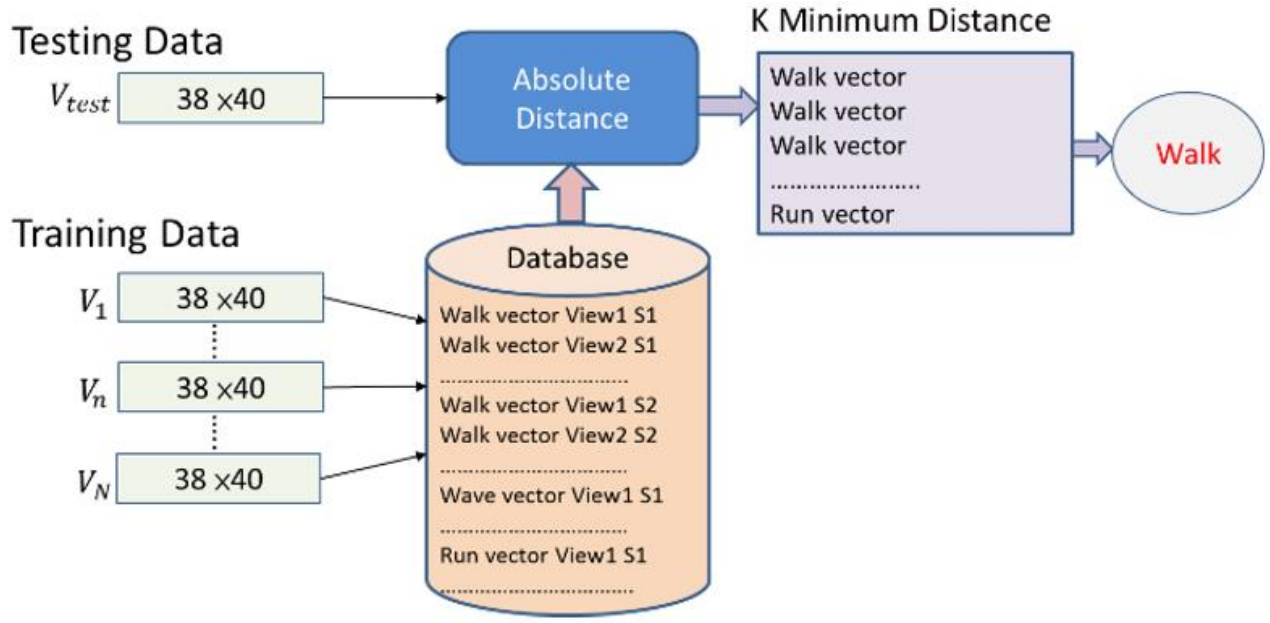


Fig. 18: Overview of NNC for the proposed work

Feature reduction and classification

Gaussian Mixture Model Classifier

3 Gaussian functions - KTH

3 Gaussian functions - WEIZMANN

4 Gaussian functions - MuHAvi

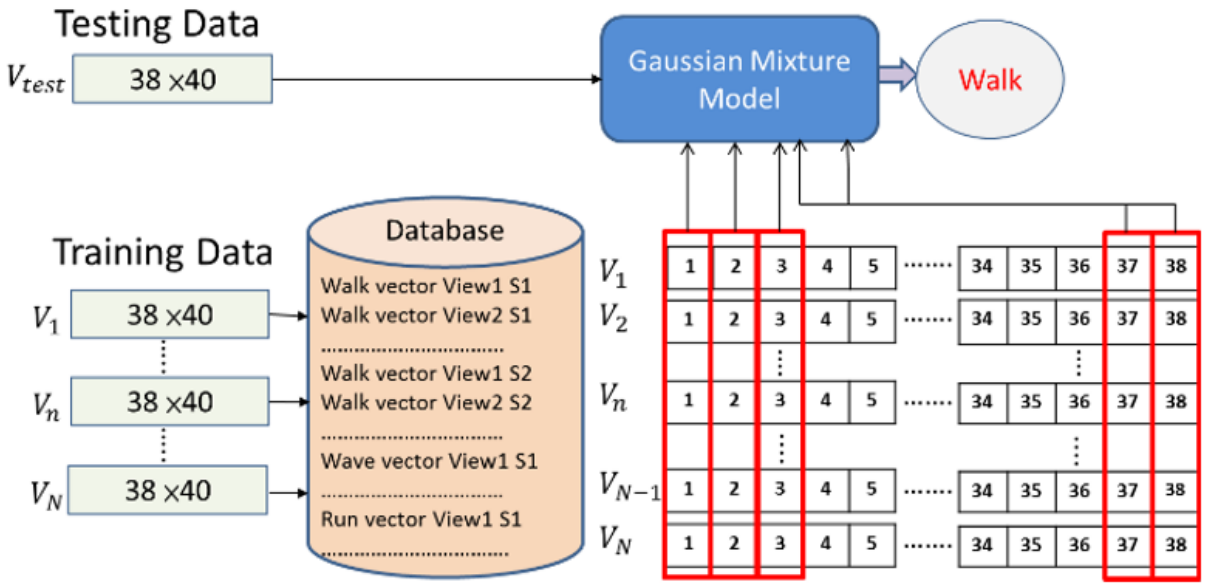


Fig. 19: Overview of GMMC for the proposed work

Feature reduction and classification

Nearest Mean Classifier

Tính mean của các các feature vectors trong cùng một action và cùng một view

Cực tiểu khoảng các tuyệt đối của testing vector và training vector

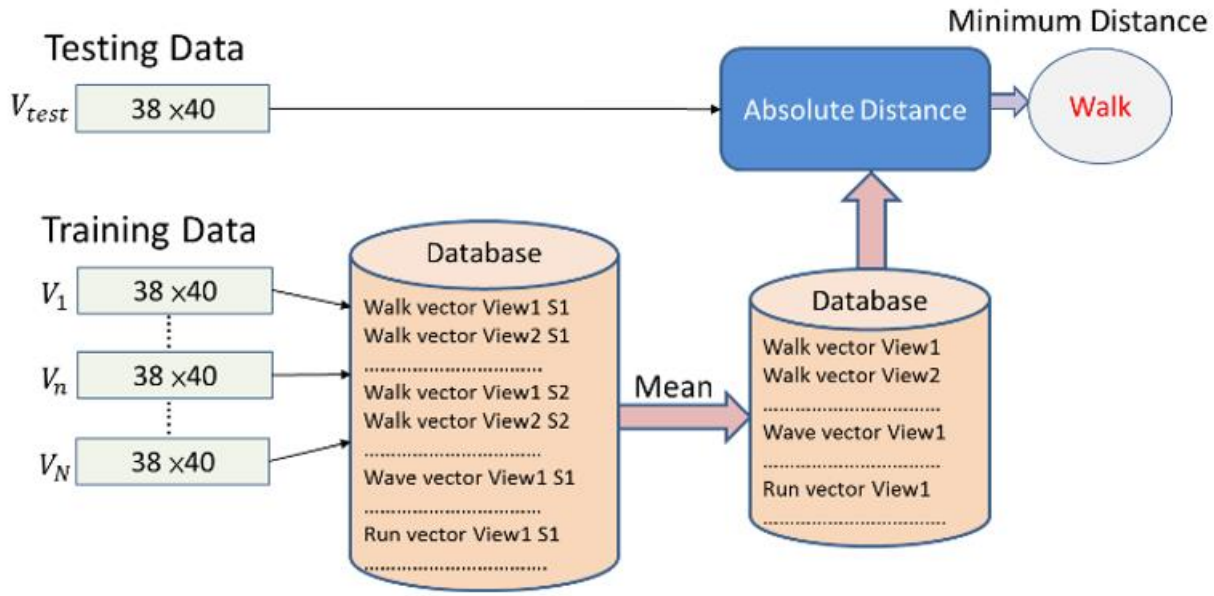


Fig. 20: Overview of NMC for the proposed work

Experiment result

Subject Invariance Evaluation

Khởi tạo:

Ns = 5

Số lượng scale: 6

- 38 features

40-bin histogram

- 1520 dimensional space

Table I Recognition performance of our approach by using NNC, GMMC, NMC

	KTH	WEIZMANN	MuHA Vi
Our approach (NNC)	89.31%	87.78%	92.50%
Our approach (GMMC)	90.21%	91.11%	93.21%
Our approach (NMC)	90.58%	95.56%	97.50%

Experiment result

Subject Invariance Evaluation

Leave-One-Out Cross-Validation (LOOCV)

GMMC và NMC tốt hơn các phương pháp trước đây với bộ dữ liệu WEIZMANN và MuHAVi dataset

Table II Comparative results on the KTH, WEIZMANN, MuHAVi datasets for subject invariance

	KTH	WEIZMANN	MuHAVi
Our approach (NMC)	90.58%	95.56%	97.50%
S. Gong et al.[18]	93.17%	96.66%	91.78%
Niebles et al. [15]	83.30%	90.00%	--
Dollar et al. [14]	81.17%	85.20%	--
Zhang et al. [23]	91.33%	92.89%	--
Gilbert et al. [21]	89.92%	--	--
Savarese et al. [22]	86.83%	--	--
Nowozin et al.[24]	84.72%	--	--

Experiment result

View Invariance Evaluation

Train trên một view và test trên các view còn lại.

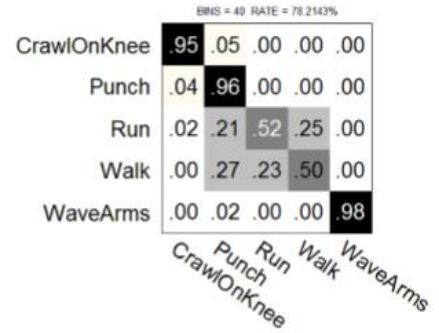
Chứng minh: phương pháp hiệu quả khi thay đổi view

Lặp lại trên cả 8 view

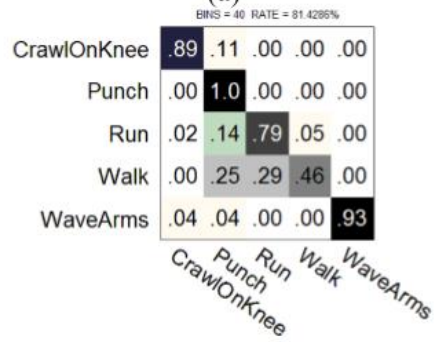
- Recognition rates

  GMMC: 78,2143%

  NMC: 81,4286%



(a)



(b)

Fig. 24: Recognition performance of view invariance evaluation using confusion matrices: (a) MuHAvi dataset using GMMC (b) MuHAvi dataset using NMC

Experiment result

View Invariance Evaluation

Train trên một view và test trên các view còn lại.

Chứng minh: phương pháp hiệu quả khi thay đổi view

Lặp lại trên cả 8 view

- Recognition rates view 3, view 5, view 6 và view 8 tốt hơn các view còn lại
- Những view trên chứa nhiều thông tin hơn

Table III Recognition performance of each view in MuHAVi dataset using GMMC and NMC

Training View	GMMC	NMC
View1	68.57%	70.36%
View2	68.93%	75.00%
View3	80.00%	83.21%
View4	82.86%	83.57%
View5	77.14%	82.86%
View6	80.71%	85.00%
View7	80.36%	80.36%
View8	80.36%	82.14%

Table IV Comparative results on the MuHAVi dataset for view invariance evaluation

	MuHAVi
Our approach (NMC)	81.43%
S. Gong et al.[18]	72.85%
A. Eweiwi et al. [25]	77.50%

Experiment result

View Invariance Evaluation

Train trên một bộ dataset này và test trên bộ dataset khác

Chứng minh: phương pháp hiệu quả khi thay đổi hướng camera, môi trường

Table V Recognition rate of the proposed approach for cross dataset testing using GMMC

Test \ Train	KTH	WEIZMANN	MuHAVi
KTH	90.22%	97.00%	93.33%
WEIZMANN	85.19%	91.11%	96.30%
MuHAVi	83.14%	84.50%	93.21%

- Khi train với bộ MuHAVi thì kết quả tốt hơn vì nó chứa nhiều view (có view của 2 bộ còn lại)

Table VI Recognition rate of the proposed approach for cross dataset testing using NMC

Test \ Train	KTH	WEIZMANN	MuHAVi
KTH	90.58%	97.00%	96.30%
WEIZMANN	84.67%	95.56%	95.56%
MuHAVi	84.53%	88.50%	97.50%

Experiment result

Auto Labeling

Sử dụng: (1/4)T-frame

Độ tương đồng: $S = \frac{F \times N_b - D}{F \times N_b}$

F: số lượng feature (38)

Nb: số lượng bin (40)

D: khoảng cách tuyệt đối giữa testing/training feature vector

- Nếu S < 70%, skip I (15) frame và tìm tiếp
- Nếu S > 70%, sử dụng (1/2)T-frame và tiếp tục quá trình.

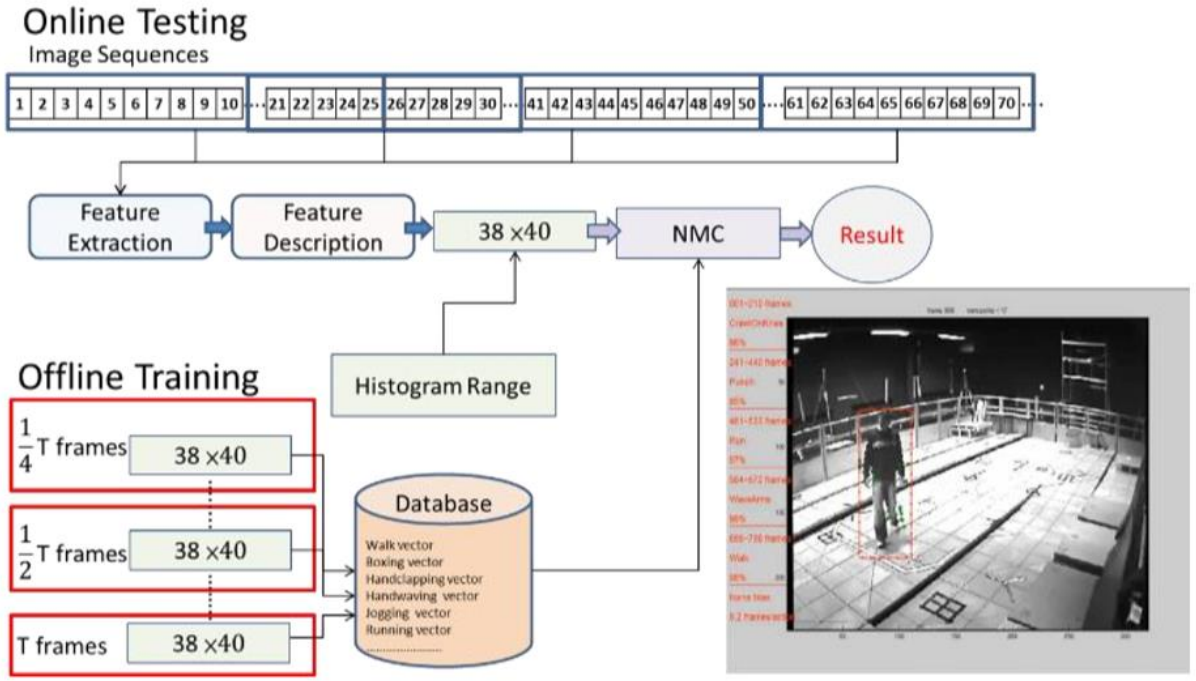


Fig. 25: Overview of auto labeling

Experiment result

Auto Labeling

Để tìm frame kết thúc action, sử dụng different rate: $R = \frac{D_c - D_l}{D_c}$

Dc là khoảnh cách tuyệt đối của feature vector giữa current scanning window và training database

DI khoảng cách tuyệt đối của feature vector giữa last scanning window và training database

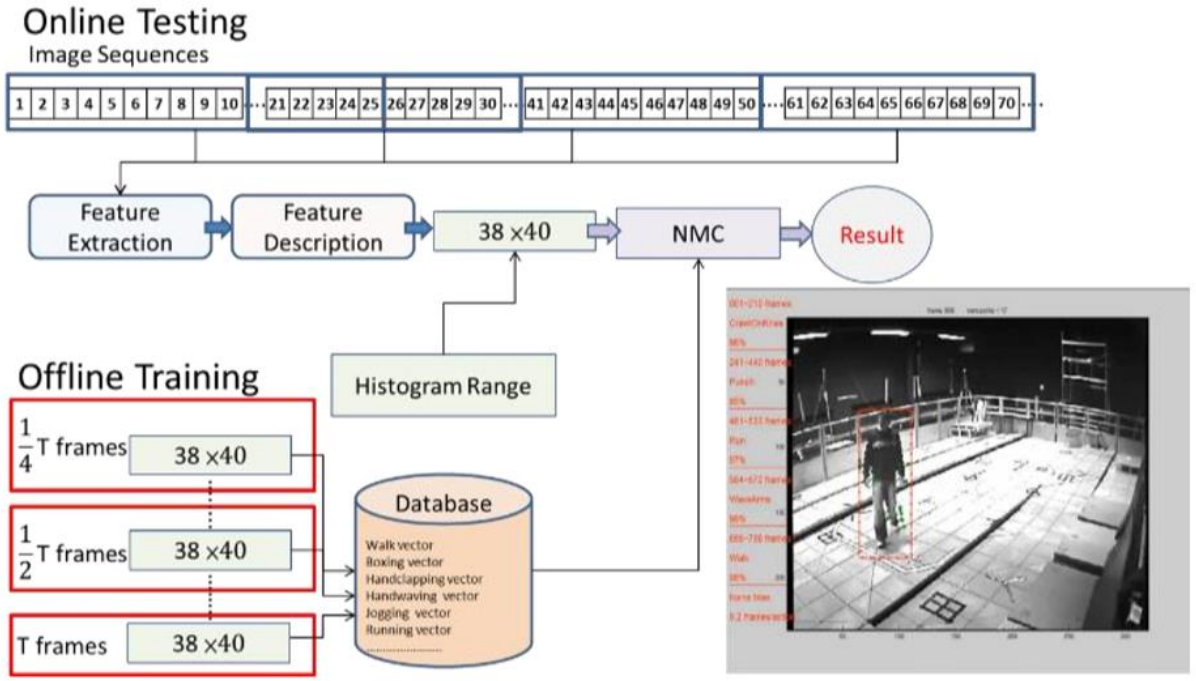


Fig. 25: Overview of auto labeling

Experiment result

Auto Labeling

Để tìm frame kết thúc action, sử dụng different rate: $R = \frac{D_C - D_l}{D_C}$

- Nếu R > 10% thì set frame hiện tại là kết thúc action.
- Gán nhãn cho hành động
- Reset quá trình từ đầu sử dụng (1/4) T frames (T = 100)

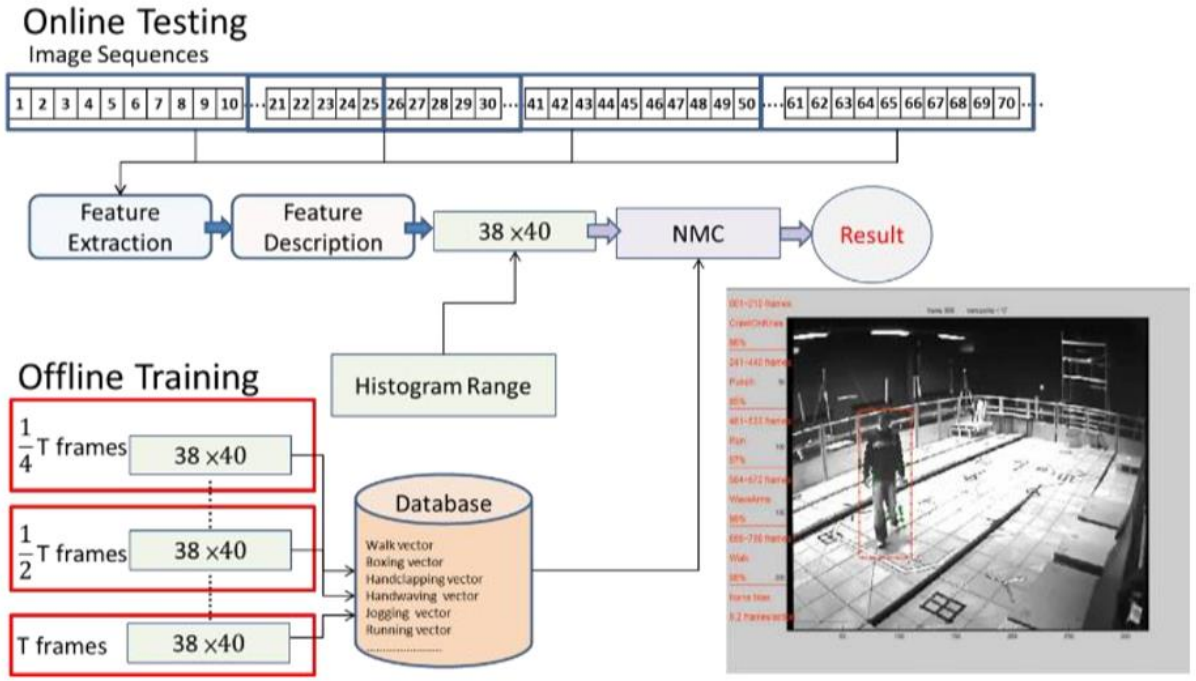


Fig. 25: Overview of auto labeling

Multi-view fusion for activity recognition using deep neural networks

Thách thức của các phương pháp truyền thống dựa trên spatio-temporal feature:

- Foreground extraction hiệu quả kém với các môi trường phức tạp
- Vị trí đặt camera thay đổi
- Các chuỗi action không được đánh dấu điểm bắt đầu và kết thúc

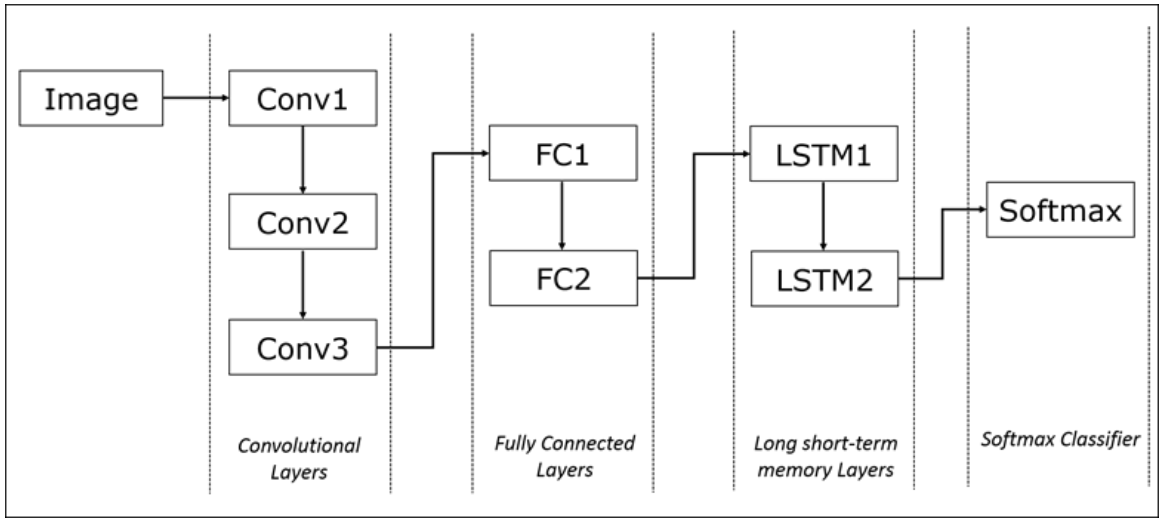
Introduction

So sánh khả năng nhận dạng của ConvNet LSTM và spatio-temporal feature

- ConvNet LSTM:
  - Đem lại hiệu quả cao hơn với một hay nhiều view
  - Fed trực tiếp các input sample thay vì phải foreground extraction

Related work

- Giải pháp: Kết hợp ConvNets và LSTMs để phân loại spatio-temporal data
- Đề xuất hai kỹ thuật fushion, thử nghiệm trên WVU dataset
- Sử dụng 2D CNN và fushion các view thay vì 3D CNN có chi phí train cao



System architecture

Building blocks of the deep neural network

Grayscale image: 640 x 480

Resize: 256 x 256

1st conv layer: 20 kernels x size 5 x 5

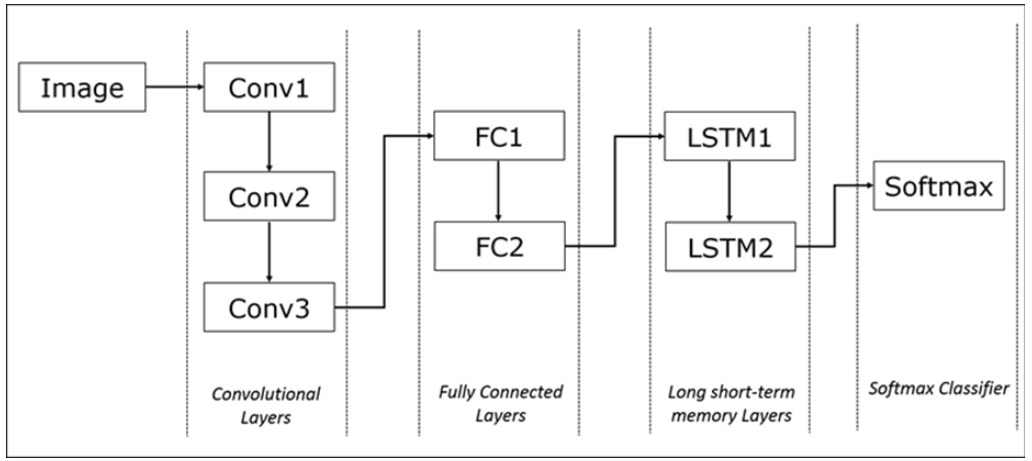
2D max pooling

2nd conv layer: 50 kernels x size 5 x 5

2D max pooling

3rd conv layer: 50 kernels x size 4 x 4

2D max pooling



System architecture

Building blocks of the deep neural network

Result: 50 channels of 29 x 29

FC1 size: 1000

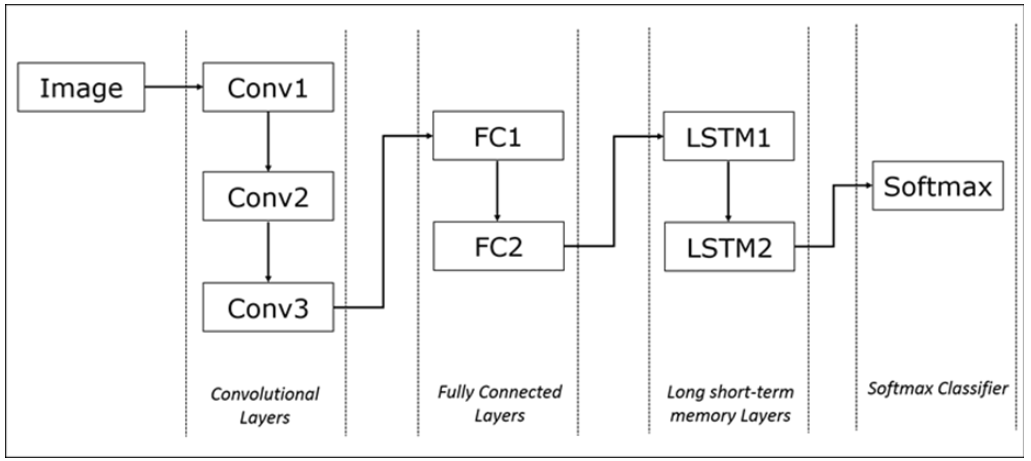
FC2 size: 500

2 x LSTM: x 1 hidden layer size 512

512 memory cell

Output: 1 x 100 vector

Softmax



System architecture

Building blocks of the deep neural network

LSTM:

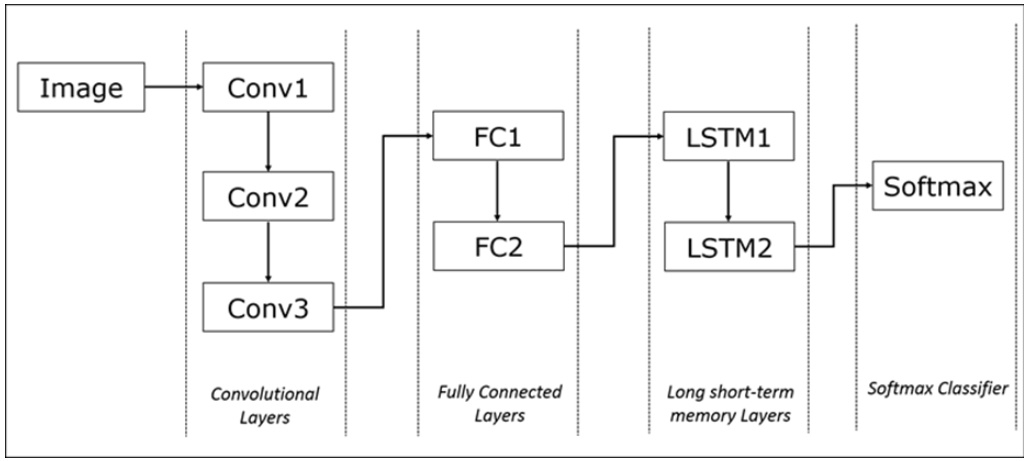
$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci}c_{t-1} + b_i)$$

$$f_t = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + W_{cf}c_{t-1} + b_f)$$

$$c_t = f_t \otimes c_{t-1} + i_t \otimes tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c)$$

$$o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + W_{co}c_t + b_o)$$

$$h_t = tanh(c_t) \otimes o_t \qquad y_t = W_{ho} \otimes h_t + b_o$$



System architecture

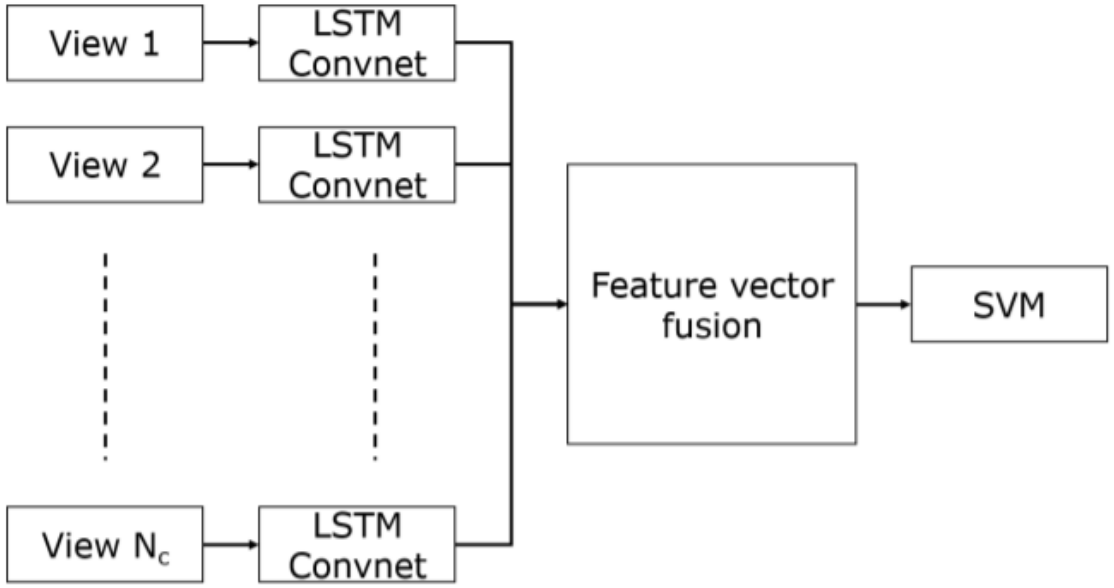
Multi-view fusion using deep neural networks

- Sử dụng ConvNet LSTMs để tạo feature:

Output mỗi view: 1 x 100 vector

Không gian feature: 1 x Nc ∗ 100

Sử dụng SVM để phân lớp các action

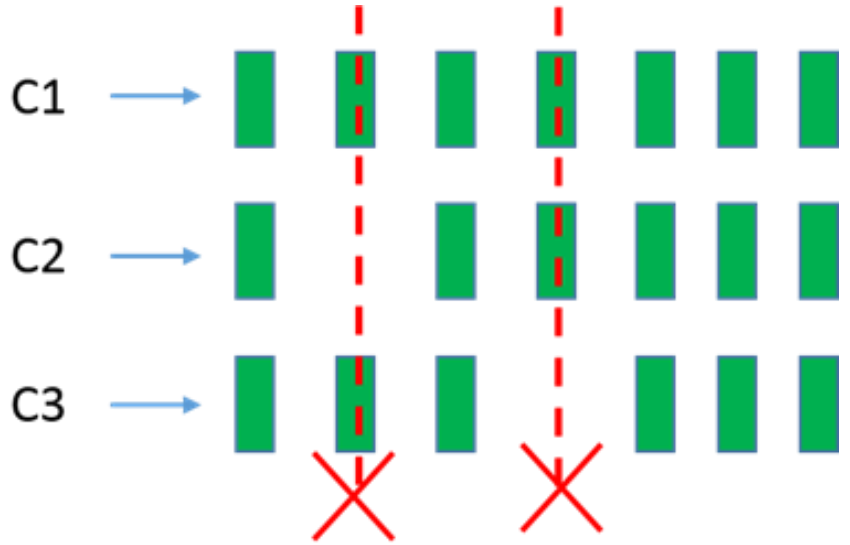


System architecture

Multi-view fusion using deep neural networks

- Sử dụng ConvNet LSTMs để tạo feature:

Nạp các frame đồng bộ theo từng view



System architecture

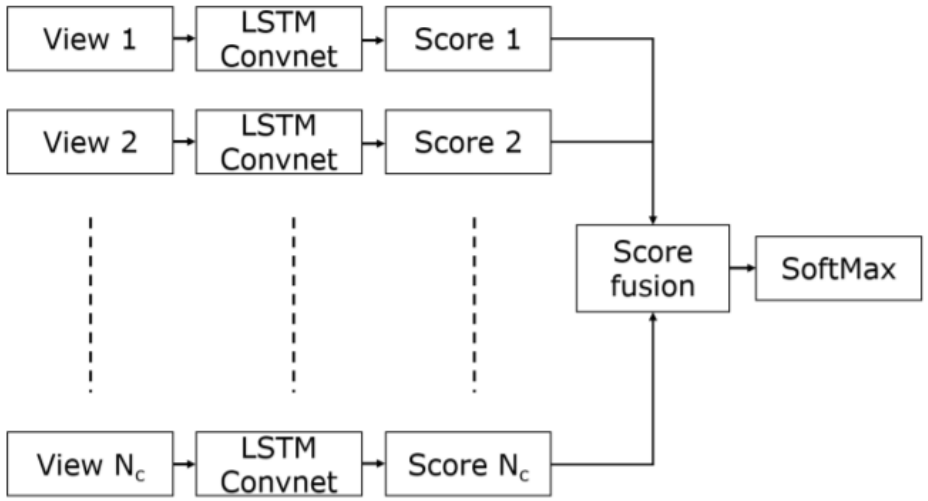
Multi-view fusion using deep neural networks

- Score fusion

  S_{av}: score của action a trong view v.

  Action a có score cao nhất sẽ được gán cho output class

$$F_a = \frac{\sum S_{av}}{N_c}$$



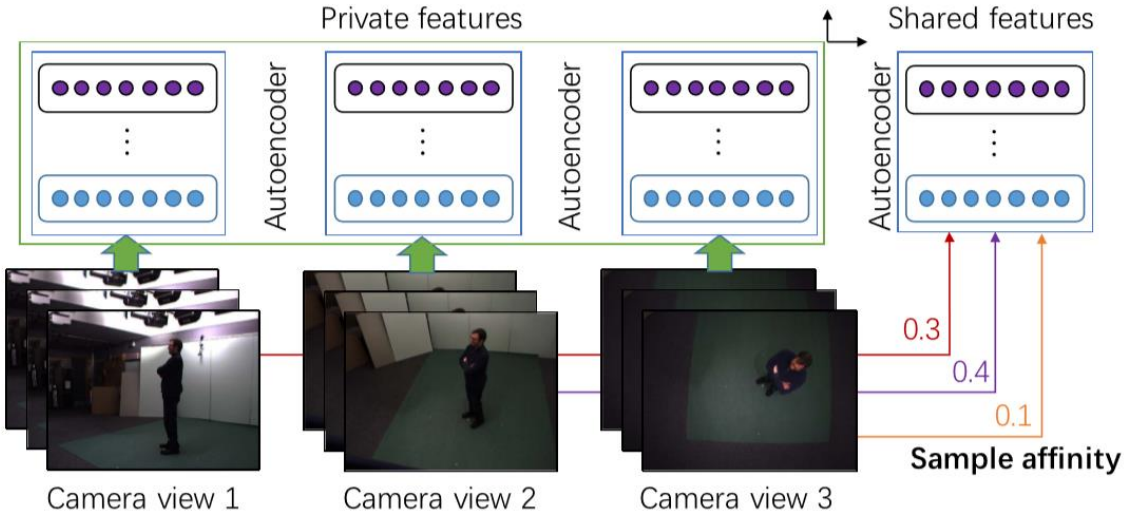


Fig. 2. Overview of the proposed method.

Introduction

Vấn đề của các phương pháp trước:

- Trọng số của các view khi xây dựng feature là như nhau

- Shared feature không thể hiện được thông tin riêng của từng view

➢ Đề xuất Sample-affinity matrix (SAM)

- View-invariant features = shared features + private features

- Tăng hiệu quả nhờ sử dụng nhãn và stack nhiều layer của features

Related Work

MvDN của Kan	Proposed approach
Compute the between-class and within-class Laplacian matrices	SAM Z directly captures with-in class between-view information and between class with-in view information
Does not encode such distance	Measures the distance between two views of the same sample

DEEPLY LEARNED VIEW-INVARIANT FEATURES

Sample-Affinity Matrix (SAM)

Training videos của V views: $\{X^v, y^v\}_{v=1}^V$

Data của view v gồm N videos: $X^v = [x_1^v, \dots, x_N^v]$

Label: $y^v = [y_1^v, \dots, y_N^v]$

$$Z = \text{diag}(Z_1, \cdots, Z_N), Z_i = \begin{pmatrix} 0 & z_i^{12} & \cdots & z_i^{1V} \\ z_i^{21} & 0 & \cdots & z_i^{2V} \\ \vdots & \vdots & \vdots & \vdots \\ z_i^{V1} & z_i^{V2} & \cdots & 0 \end{pmatrix}$$

$z_i^{uv} = \exp(\|x_i^v - x_i^u\|^2/2c)$ parameterized bởi c: khoảng cách giữa 2 view của sample thứ i

DEEPLY LEARNED VIEW-INVARIANT FEATURES

Sample-Affinity Matrix (SAM)

Bản chất của SAM Z: captures thông tin within-class between-view và between-class within-view information

- Mỗi block Zi thể hiện sự khác nhau giữa các view trong cùng một class

➢ Thể hiện sự thay đổi của action khi đổi view

- Các block không phải đường chéo trong SAM Z đều 0 giúp hạn chế thông tin sharing giữa các class trong cùng một view

➢ Feature cùng view nhưng khác class là dễ phân biệt

DEEPLY LEARNED VIEW-INVARIANT FEATURES

Preliminary on Autoencoders

NORMAL AUTOENCODER

"encoder" f1(·): h = f1(x)

"decoder" f2(·): o = f2(h)

objective function:

min $\sum_{i=1}^{N} \|\mathbf{x}_i - \mathbf{f}_2(\mathbf{f}_1(\mathbf{x}_i))\|^2$

MARGINALIZED STACKED DENOISING AUTOENCODER (MSDA)

$$\min \sum_{i=1}^{N} \|\mathbf{x}_i - W\tilde{\mathbf{x}}_i\|^2$$

Trong đó:

W: single mapping

$\sim x_i$: corrupted version của x_i bằng cách set mỗi feature về 0 với xác xuất p

mSDA thực hiện m pass trên training set với nhiều corruption khác nhau (một kiểu dropout)

$m \to \infty$: học được W hiệu quả với noise

DEEPLY LEARNED VIEW-INVARIANT FEATURES

Single-Layer Feature Learning

Objective function: $\min_{W,\{G^v\}} \mathcal{Q}, \quad \mathcal{Q} = \|W\tilde{X} - XZ\|_F^2 + \sum_v \Big[\alpha\|G^v\tilde{X}^v - X^v\|_F^2 + \beta\|W^{\mathrm{T}}G^v\|_F^2 + \gamma\,\mathrm{Tr}(P^vX^vLX^{v\mathrm{T}}P^{v\mathrm{T}})\Big]$

W: mapping matrix cho shared features

$\{G^v\}_{v=1}^V$: mapping matrix cho private features

4 term + 3 parameters:

- Học shared features giữa các view
- Học view-specific private features
- 2 x Model regularizers

DEEPLY LEARNED VIEW-INVARIANT FEATURES

Shared Features

Hàm tính sự khác biệt giữa target view v với toàn bộ các view:

$$\psi = \sum_{i=1}^{N} \sum_{v=1}^{V} \| W\tilde{\mathbf{x}}_i^v - \sum_u \mathbf{x}_i^u z_i^{uv} \|^2 = \| W\tilde{X} - XZ \|_F^2$$

z: trọng số thể hiện mức độ ảnh hưởng của action trong view u với sample x trong view v

Z: SAM encoding các z

W: single linear mapping cho corrupted input ~x

~X: corrupted input của X (dropout)

DEEPLY LEARNED VIEW-INVARIANT FEATURES

Shared Features

Sử dụng trọng số z cho các view

SAM Z giới hạn thông tin share giữa các sample (ma trận đường chéo) do đó không capture được view-invariant information cho cross view action recognition.

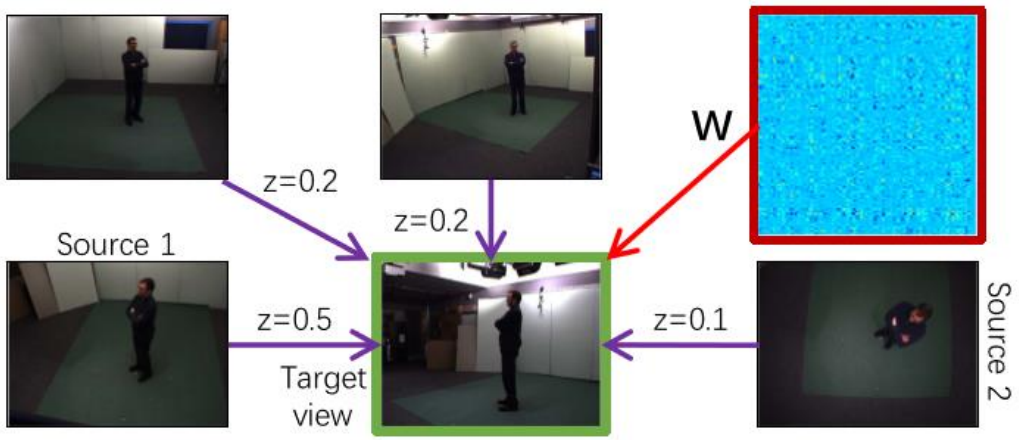


Fig. 3. Learning shared features using weighted samples.

DEEPLY LEARNED VIEW-INVARIANT FEATURES

Private Features

Thông tin phân biệt giữa các view

Học view-specific private features cho các sample trong view thứ v sử dụng mapping matrix G

$$\phi_v = \sum_{i=1}^{N} \|G^v \tilde{\mathbf{x}}_i^v - \mathbf{x}_i^v\|^2 = \|G^v \tilde{X}^v - X^v\|_F^2$$

Hàm trên có thể tính cả các thông tin share view dư thừa => Cần loại bỏ bằng hàm phạt (tăng sự không thống nhất giữa hai ma trận W và G):

$$r_{1v} = \|W^{\mathbf{T}} G^v\|_F^2$$

DEEPLY LEARNED VIEW-INVARIANT FEATURES

Label Information

Tăng tính phân biệt của các feature:

Hiệu chỉnh shared và private features của cùng view và cùng class phải giống nhau

- Hàm phạt (within-class within-view) đối với mapping matrix W và G

$$
\begin{aligned}
r_{2v} &= \sum_{i=1}^{N}\sum_{j=1}^{N}\left[\|W\mathbf{x}_i^v - W\mathbf{x}_j^v\|^2 + \|G^v\mathbf{x}_i^v - G^v\mathbf{x}_j^v\|^2\right] \\
&= \mathrm{Tr}(WX^vLX^{v\mathrm{T}}W^{\mathrm{T}}) + \mathrm{Tr}(G^vX^vLX^{v\mathrm{T}}G^{v\mathrm{T}}) \\
&= \mathrm{Tr}(P^vX^vLX^{v\mathrm{T}}P^{v\mathrm{T}}),
\end{aligned}
$$

DEEPLY LEARNED VIEW-INVARIANT FEATURES

Label Information

L = D-A: label view Laplacian matrix

Quan hệ giữa 2 sample: $a_{(i,j)} = 1\ nếu\ y_i = y_j\ (0\ nếu\ ngược\ lại)$

D: ma trận chéo $D_{(i,i)} = \sum_{j=1}^{N} a_{(i,j)}$

A: ma trận kề với các phần tử $a_{(i,j)}$

$$
\begin{aligned}
r_{2v} &= \sum_{i=1}^{N}\sum_{j=1}^{N}\left[\|W\mathbf{x}_i^v - W\mathbf{x}_j^v\|^2 + \|G^v\mathbf{x}_i^v - G^v\mathbf{x}_j^v\|^2\right] \\
&= \mathrm{Tr}(WX^vLX^{v\mathrm{T}}W^{\mathrm{T}}) + \mathrm{Tr}(G^vX^vLX^{v\mathrm{T}}G^{v\mathrm{T}}) \\
&= \mathrm{Tr}(P^vX^vLX^{v\mathrm{T}}P^{v\mathrm{T}}),
\end{aligned}
$$

DEEPLY LEARNED VIEW-INVARIANT FEATURES

Deep Architecture

Deep model bằng các stack nhiều tầng feature learners

Algorithm 1 Learning view-invariant features

Input: $\{(\mathbf{x}_i^v, y_i)\}_{i=1,v=1}^{N,V}$.
Output: $\{W_k\}_{k=1}^{K}$,$\{G_k^v\}_{k=1,v=1}^{K,V}$.
for Layer $k = 1$ to K **do**
    Input $H_{(k-1)w}$ for learning W_k.
    Input $H_{(k-1)g}^v$ for learning G_k^v.
    while not converge **do**
        Update W_k using Eq. (6);
        Update $\{G_k^v\}_{v=1}^{V}$ using Eq. (7);
    end while
    Compute H_{kw} by: $H_{kw} = \sigma(W_k H_{(k-1)w})$.
    Compute $\{H_{kg}^v\}_{v=1}^{V}$ by: $H_{kg}^v = \sigma(G_k^v H_{(k-1)g}^v)$.
end for

My Experience

- Kỹ thuật tạo View-Invariant Features sử dụng mạng neural đem lại hiệu quả cao nhất
- Khi xây dựng feature cần chú ý trọng số giữa các view
- Mạng neural trong các phương pháp trên đều là FC với số layer thấp (so với các kiến trúc mạng như VGG, Google net, …)
- Các frame của video được feed lần lượt vào mạng => không tách được temporal information