HUMAN POSE STREAM FOR MULTI-STREAM CONVOLUTIONAL NETWORK IN VIDEO ACTION CLASSIFICATION

BY

JACKSON TAN ZHE SHENG

A REPORT

SUBMITTED TO

Universiti Tunku Abdul Rahman

in partial fulfillment of the requirements

for the degree of

BACHELOR OF COMPUTER SCIENCE (HONS)

Faculty of Information and Communication Technology
(Perak Campus)

January 2018

UNIVERSITI TUNKU ABDUL RAHMAN

REPORT STATUS DECLARATION FORM

Title: ____

____

____

Academic Session: _____________

I ____

(CAPITAL LETTER)

declare that I allow this Final Year Project Report to be kept in

Universiti Tunku Abdul Rahman Library subject to the regulations as follows:

1. The dissertation is a property of the Library.
2. The Library is allowed to make copies of this dissertation for academic purposes.

Verified by,

_______________________	_______________________
(Author’s signature)	(Supervisor’s signature)

Address:

____

________________________ ________________________

________________________ Supervisor’s name

Date: ___________________ **Date**: ___________________

HUMAN POSE STREAM FOR MULTI-STREAM CONVOLUTIONAL NETWORK IN VIDEO ACTION CLASSIFICATION

BY

JACKSON TAN ZHE SHENG

A REPORT
SUBMITTED TO
Universiti Tunku Abdul Rahman
in partial fulfillment of the requirements
for the degree of
BACHELOR OF COMPUTER SCIENCE (HONS)
Faculty of Information and Communication Technology
(Perak Campus)

January 2018

DECLARATION OF ORIGINALITY

I declare that this report entitled "HUMAN POSE STREAM FOR MULTI-STREAM CONVOLUTIONAL NETWORK IN VIDEO ACTION CLASSIFICATION" is my own work except as cited in the references. The report has not been accepted for any degree and is not being submitted concurrently in candidature for any degree or other award.

Signature : ______________________

Name : ______________________

Date : ______________________

ACKNOWLEDGEMENT

I would like to express my gratitude to my supervisors, Dr. Hung Khoon who has guided me throughout this project and provided me with the chance to set involved in this deep learning video processing project. This project prepares me well for a career in the deep learning field.

Finally, I would like to thank my friends, parents and family for their continuous love, support and encouragement throughout my studies. Without them, I would not be able to complete my course and this project. Thanks.

ABSTRACT

Multi-stream Convolutional neural network (ConvNet/ CNN) has been shown to deliver impressive performance for video processing tasks such as the human action classification. In previous studies, multi-stream architecture has included various types of modalities have been introduced in order to utilize the information which embedded in the video.

In this work, we propose a multi-stream architecture which combined the spatial stream (still video frames) with a novel human pose stream. The input of the human pose stream consists of multiple human pose frames without background noises that stacked together which computed through a deep human posture estimation model (OpenPose). Our pilot study shows that the performance in terms of accuracy of our proposed system which fuses the spatial stream and human pose stream (91.3%) out-performs both the spatial (87.1%) or human pose stream (71.0%) when considered separately on the UCF-101 dataset (26 classes).

TABLE OF CONTENTS

TITLE **i**

DECLARATION OF ORIGINALITY **ii**

ACKNOWLEDGEMENT **iii**

ABSTRACT **iv**

TABLE OF CONTENTS **v**

LIST OF TABLES **vii**

LIST OF FIGURES **viii**

LIST OF ABBREVIATIONS **x**

Chapter 1: Introduction **1**

1.1 Project Scope and Objectives 2

1.2 Report organization 2

Chapter 2: Literature Review **3**

2.1 Overview of the related works 3

2.2 Various modalities of video 3
2.2.1 Raw video modality 3
2.2.2 Raw frame modality 4
2.2.3 Motion modality 5
2.2.4 Audio modality 5
2.2.5 RGB differences and warped optical flow modalities 6
2.2.6 Motion vector modality 7

2.3 Human pose modality 8

Chapter 3: Multi-stream architecture for video classification **10**

3,1 CNN configuration 11

3.2 Spatial Stream CNN 11

3.3 Human Pose Stream CNN 14

3.4 Fusing the 2 streams 16

Chapter 4: Implementation details of the proposed system **18**

4.1 Overall design specification 18

4.2 Data Preprocessing and Data Augmentation 18
4.3 Training and validation setup 20
4.4.5 Testing setup 20

Chapter 5: Evaluation **22**

5.1 Datasets used and evaluation protocol 22

5.2 Discussion of the result 23

5.3 Transfer learning with pre-trained ResNet-50 model in CNNs 28
5.3.1 Spatial CNN 28
5.3.2 Human pose CNN 30

5.4 Average fusion of 2-stream CNN 31

Chapter 6: Conclusion **33**

6.1 Discussion 33

6.2 Future works 33

Bibliography **34**

Poster **42**

Turnitin Similarity Report **43**

CHECKLIST FOR FYP2 THESIS SUBMISSION **45**

LIST OF TABLES

Table Number	Title	Page
Table 4.1	Training time for each stream	20
Table 5.1	10-cropped testing results	23

LIST OF FIGURES

Figure Number	Title	Page
Figure 1	Video action classification	1
Figure 2.1	Overview of 2 steps neural recognition scheme	3
Figure 2.2	Multi-resolution CNN	4
Figure 2.3	Two-stream architecture for video classification	5
Figure 2.4	3-stream video classification with adaptive fusion	6
Figure 2.5	RGB frames, RGB differences, optical flow, warped optical flow	7
Figure 2.6	Motion vector vs optical flow	8
Figure 2.7	Input and output of the OpenPose	9
Figure 2.8	OpenPose Results on the MPII datas	9
Figure 3.1	Overview of the full architecture	10
Figure 3.2	Comparison between the Residual Network and other architectures	11
Figure 3.3	Transfer learning in spatial stream	13
Figure 3.4	Video frame to Human pose frame	14
Figure 3.5	Transfer learning human pose stream	16
Figure 3.5	Overview of the proposed multi-stream network	17
Figure 4.1	Spatial stream input preprocessing	18
Figure 4.2	Temporal stream input preprocessing	19
Figure 4.3	10-cropped testing samples	21
Figure 5.1	Number of videos in testing set	22
Figure 5.2	Classification report for spatial stream	23
Figure 5.3	Classification report for human pose stream	24
Figure 5.4	Classification report for 2 stream CNN	25
Figure 5.5	Precision of spatial, human pose and 2 stream	26

Figure 5.6 Recall of spatial, human pose and 2 stream 26

Figure 5.7 F1-score of spatial, human pose and 2 stream 27

Figure 5.8 Raw frame and human pose frame for Archery action (g01_c02) 27

Figure 5.9 Raw frame and human pose frame for Archery action (g03_c06) 28

Figure 5.3.1 Trained from scratch VS Pre-trained model 29

Figure 5.3.2 Human pose CNN training, validation loss and accuracy 30

Figure 5.3.3 2 stream CNN training, validation loss and accuracy 31

LIST OF ABBREVIATIONS

ConvNet/ CNN	**Convolutional Neural Network**
RNN	**Recurrent Neural Network**
LSTM	**Long Short-Term Memory**
GPU	**Graphical Processing Unit**
ReLU	**Rectified Linear Unit**
2D	**2-Dimension**
3D	**3-Dimension**
ILSVRC	**ImageNet Large Scale Visual Recognition Challenge**

Chapter 1: Introduction

Multi-stream convolutional neural networks have achieved the state-of-the-art performance for the video processing tasks like human action recognition which classify a given video to a particular action.

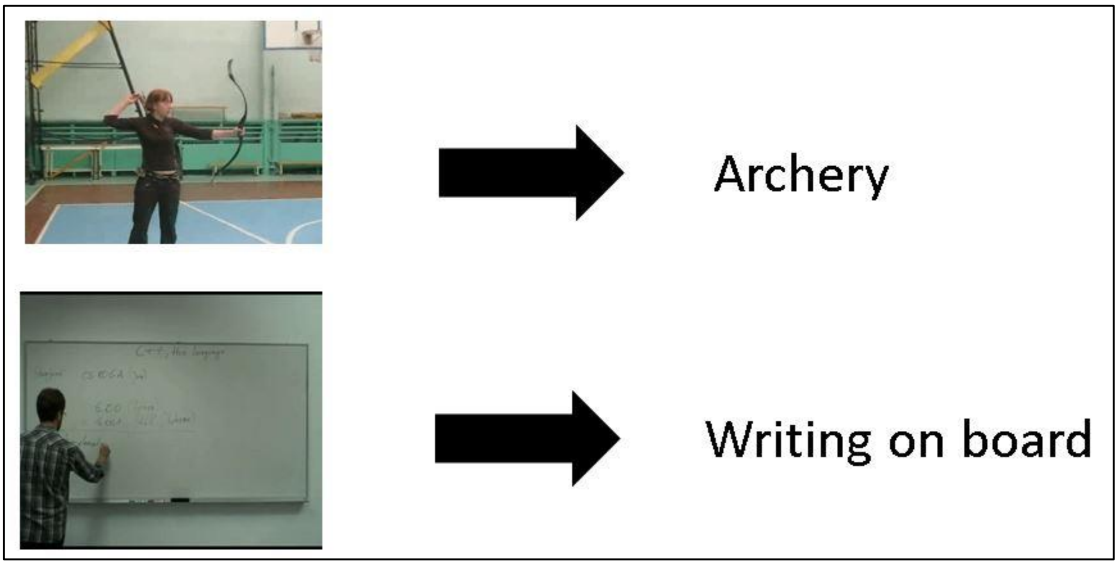


Figure 1: Video action classification

Unlike an image which only consist of the static visual information, a video is a multi-modal document that included multiple components such as the spatial, temporal, audio and etc. which need to be exploit and utilize through a multi-stream architecture.

In the past, several works have explored the various types of modalities within the video. For example, the proposed architecture by Baccouche et al. (2011) extracted the spatial-temporal information from sub-segmented raw input video. Karparthy et al. (2014) have introduced a multiple stream that works on images at different resolutions. Simonyan & Zisserman (2014) proposed a 2-stream architecture to extract the spatial and temporal information from raw video frames and multiple stacked optical flows. Wu et al. (2015) proposed a 3-stream architecture which included the spatial, motion and audio. Wang et al. (2016) introduced 2 new modalities, RGB differences and warped optical flow. Zhang et al. (2016) has proposed a new modality which was the motion vector.

Chapter 1: Introduction

The information of the human posture within each individual frames have not been utilized before in any previous works. Modality such as the optical flows contain the motion information of both the foreground and background which might serve as the noises which will distract the network. In our proposed architecture, the noises are reduced by using stacked human pose frames with the background removed in order to let our network to extract only the relevant information from the foreground through a series of human pose when a particular action is carried out.

1.1 Project Scope and Objectives

The proposed system from this project aim to recognize and classify the human actions within the input video and propose a new modality for the multi-stream CNN. The proposed architecture consists of spatial and human pose streams which are trained by using the raw video frames and stacked human pose frames respectively. Then, average fusion is used to combine both streams.

1.2 Report organization

The report is divided into 6 chapters. Chapter 1 is the introduction of the overall project as well as the project scope and objectives. In chapter 2, different types of the modalities are review and discuss which proposed in the past researches.

In chapter 3 describe our overall proposed architecture and methods to initialize the architectures through transfer learning in details. After that, the chapter 4 explain the data pre-processing for both the training and testing phases. We report our experiment results in chapter 5.

The last chapter provide the conclusion for the overall project as well as stated some future improvements.

Chapter 2: Literature Review

2.1 Overview of the related works

The literature review is organized into 2 sections. Section 2.2 discusses about various types of modalities used in several recent works. The Section 2.3 reviews a work which is used in our proposed system.

2.2 Various modalities of video

2.2.1 Raw video modality

Although CNN has achieved good performance in the image domain, initial attempts applied CNN to the video domain were less successful due to lack of sufficient training samples. Different from an image, a video contains temporal and other modalities. Baccouche et al. (2011) proposed a 2-step scheme to implement a fully automated deep model for human action classification. First, a 3D-ConvNet is used to discover the spatio-temporal features in the video. Then, a LSTM network learns the temporal pattern from the sequence of features learnt from the 3D-ConvNet. However, the proposed system was conducted on KTH dataset which contained only 600 of videos with 6 different actions.

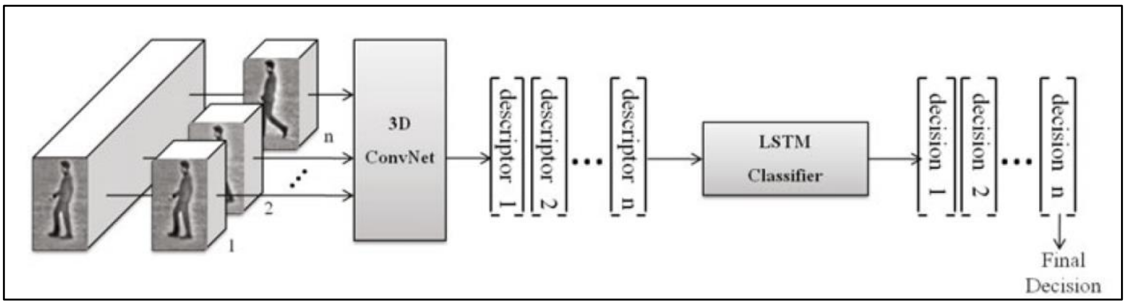


Figure 2.1: Overview of 2 steps neural recognition scheme (Baccouche et al., 2011)

2.2.2 Raw frame modality

Karpathy et al. (2014) introduced a multi-stream CNN with different resolutions which able to speed up the running time without the affecting the performance.

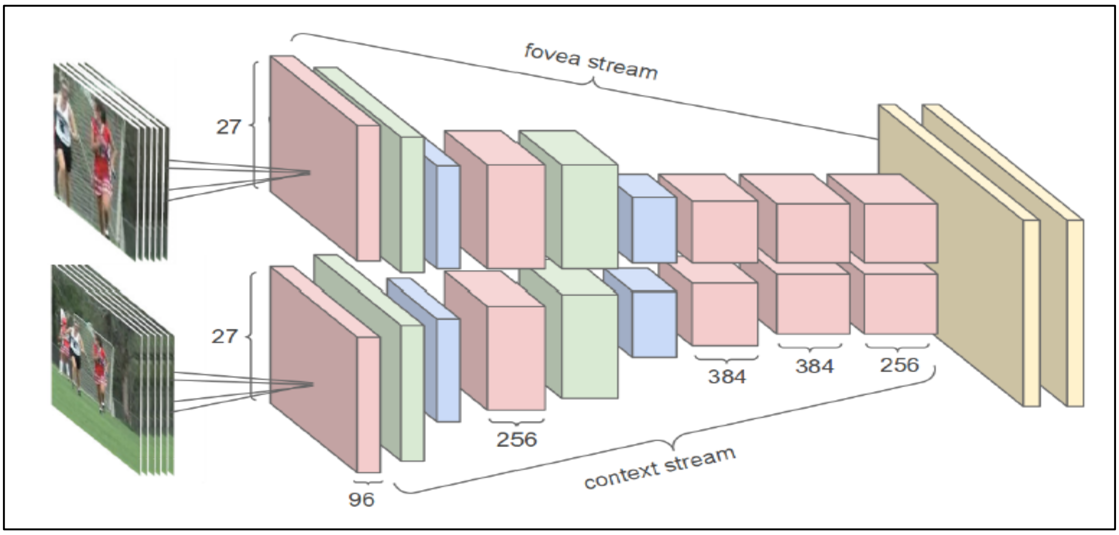


Figure 2.2: Multiresolution CNN (Karpathy et al., 2014)

Figure 2.2 shows their proposed framework. The multiresolution CNN architecture is divided into 2 streams: fovea and context stream. The fovea stream receives a low-resolution video frames as the totality of the frame whereas the context stream captures the critical part in the video which is assumed to be focused on the center region. More importantly, the resolution of the input to both stream reduce running time but it will require more computational resources which is not practical in our use case.

In addition, the authors also introduced a new dataset (Sports-1M) which contained 1 million of YouTube videos with 487 classes.

2.2.3 Motion modality

Figure 2.2.3 shows the proposed system by Simonyan & Zisserman (2014). It is a 2-stream convolutional network which incorporates spatial and temporal networks that mimic the human eyes. The spatial stream processed the raw frames of the video whereas the temporal stream was required to extract short-term motion information from the stacked optical flow which is computed from two consequence frames. The optical flow frames are pre-generated although it has a short computational time of 0.06s to avoid bottleneck.

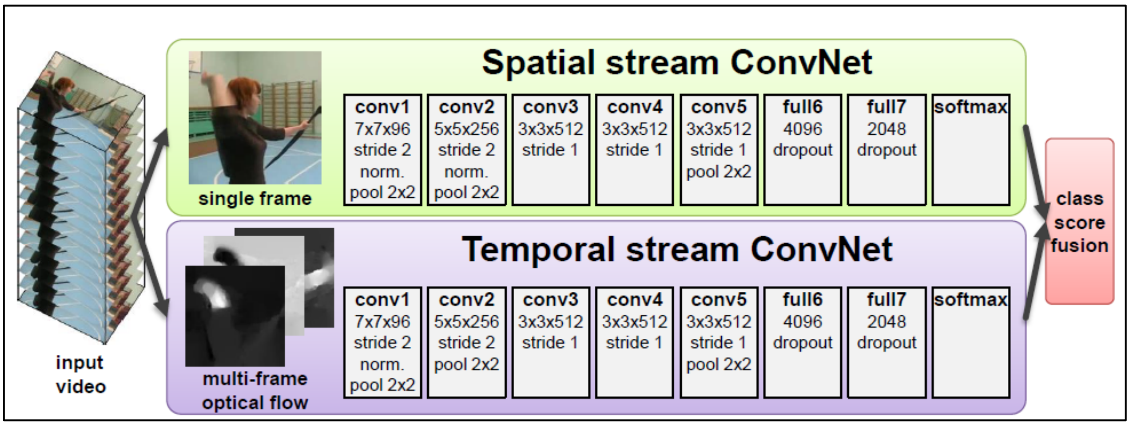


Figure 2.3: Two-stream architecture for video classification (Simonyan & Zisserman, 2014)

Their spatial stream CNN is pre-trained under ImageNet dataset which shown significant improvement over the approach of training from scratch.

In order to avoid overfitting issue for the temporal stream, they applied multi-task learning to alleviate the issue of insufficient of the training set for the temporal CNN by combining multiple datasets (UCF-101 and HMDB-51 datasets) which will end up with a better model and higher accuracy but also a higher computational cost.

2.2.4 Audio modality

Wu et al. (2015) proposed a multi-stream architecture by applied 3 CNNs for extracting the spatial, short-term temporal information and audio individually as well as 2 LSTMs for long-term motion information processing. The prediction outputs of the given deep models are combined by using the adaptive fusion methods which achieved a significant improvement compared to the previous approaches (without the audio stream) as the most of the sample videos within the UCF-101 datasets don't include the sounds.

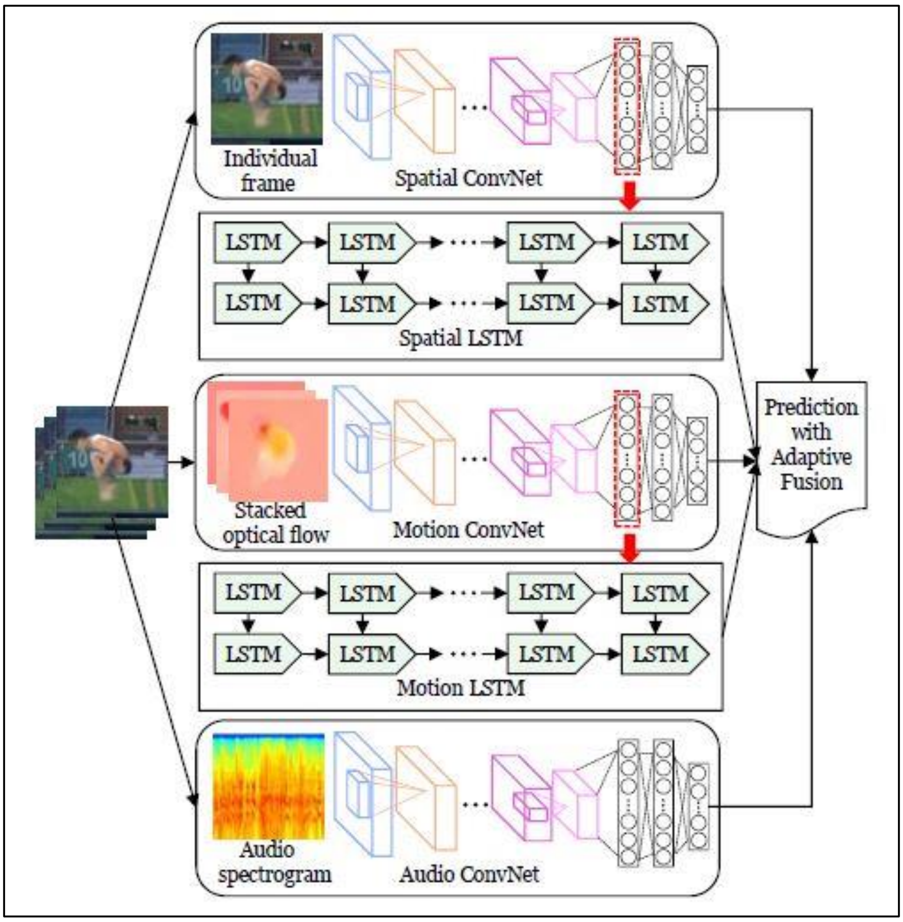


Figure 2.4: 3-stream video classification with adaptive fusion (Wu et al., 2015)

2.2.5 RGB differences and warped optical flow modalities

Other modalities have also been investigated in some other works. Wang et al. (2016) proposed newly the RGB difference and warped optical flow. The RGB difference computed from the 2 consecutive frames which indicated the changes within the spatial spaces while the warped optical flow focused on the motion of the human in the foreground which has the similar intention compared to our proposed modality.

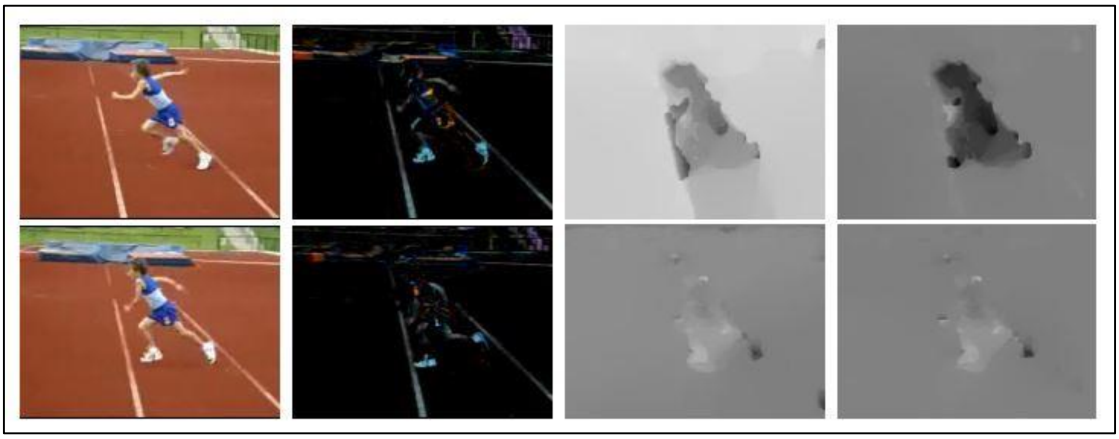


Figure 2.5: RGB frames, RGB differences, optical flow, warped optical flow (Wang et al., 2016)

According to Wang et al. (2016), they initialize the temporal stream CNN by using the weights of the spatial stream with a slight modification of the first convolutional layer due to different volume of the inputs for both streams (still video frames & stacked optical flow frames) has reported to be beneficial by avoid overfitting issue which is similar to our approach to initialize our human pose stream network.

2.2.6 Motion vector modality

Zhang et al. (2016) proposed the motion vector modality which can be get from the compressed video directly in real-time and it is less computational costly than the optical flow which required to be computed.

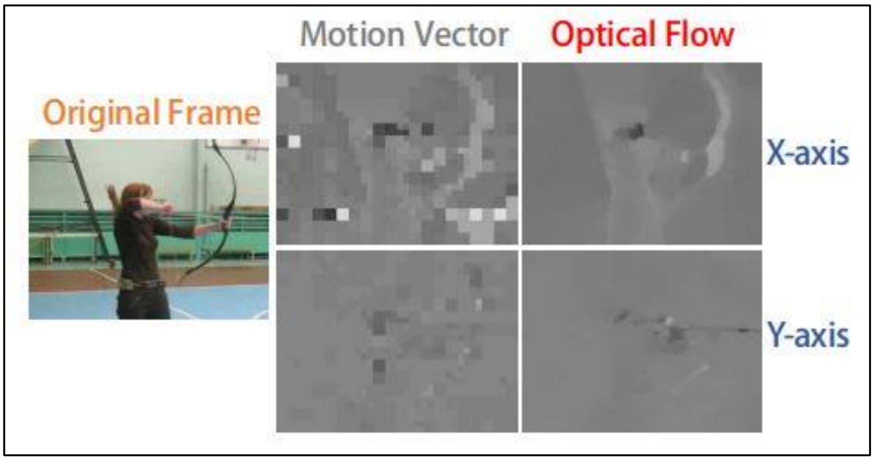


Figure 2.6: Motion vector vs optical flow (Zhang et al., 2016)

The modality of the motion vector includes a lot of noises such as its coarse structure and inaccurate motion patterns which might affect the classification performance. (Zhang et al. ,2016) This can be resolved by using transfer learning across different modalities where the pre-trained optical flow CNN was used to initialized the motion vector CNN which shown in Figure 2.6. This can be done in 3 different configurations namely the Teacher Initialization, Supervision Transfer and Combination where all 3 showed improvement compared to training from scratch.

2.3 Human pose modality

Simon et al. (2017) introduced a bottom-up approach for the multiple human posture estimation by using the Part Affinity Fields (PAFs) which known as the OpenPose. Human pose estimation is used to predict the human posture from the given input video frames.

The human pose frames are being generated by using the architecture that stated in Figure 2.7. The features, F are extracted from each raw video frame by using from the first 10 layers of the pre-trained VGG-19 model before inputted to the multi-stage CNN with different branches to further computed the part confidence maps and part affinity fields at the initial stage. The outputs of the initial stage are concatenated with the input image in order to act as a guideline for the prediction in the upcoming stages.

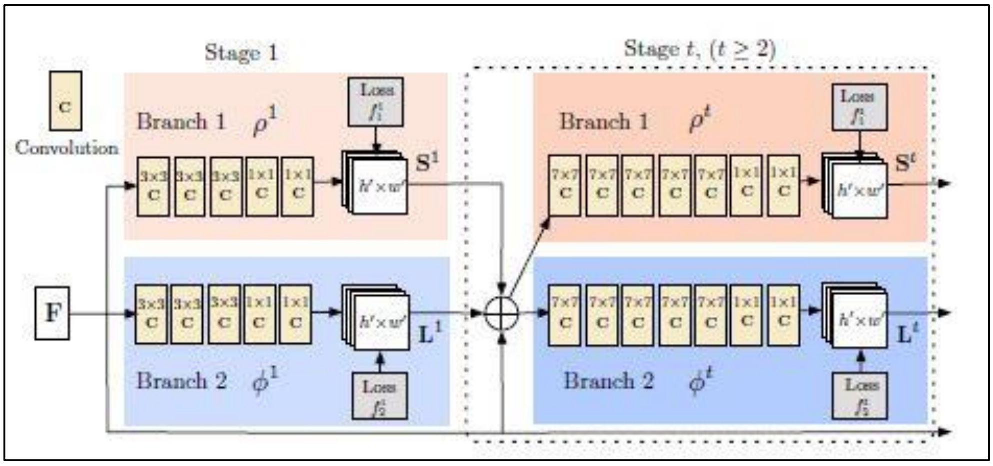
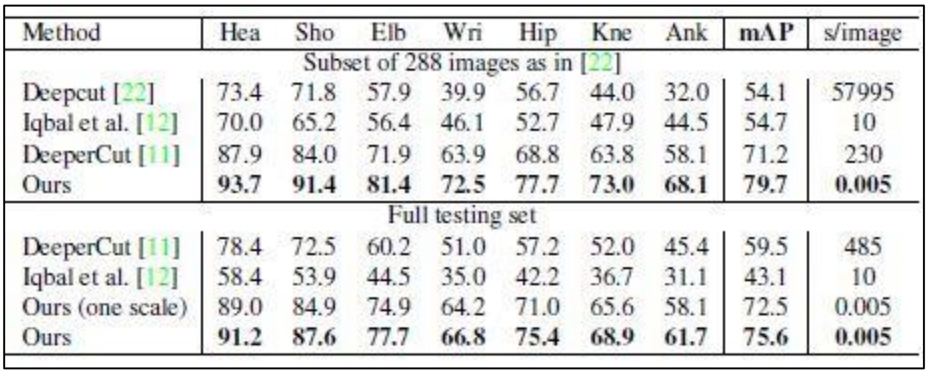


Figure 2.7: OpenPose Architecture (Simon et al., 2017)

Their proposed method is able to produce a much significant improved results compared to previous works in term of higher accuracy and lower computational time which shown in Figure 2.8. In this project, OpenPose is employed as the generative network which provided the inputs for our human pose stream.



Method	Hea	Sho	Elb	Wri	Hip	Kne	Ank	mAP	s/image
Subset of 288 images as in [22]									
Deepcut [22]	73.4	71.8	57.9	39.9	56.7	44.0	32.0	54.1	57995
Iqbal et al. [12]	70.0	65.2	56.4	46.1	52.7	47.9	44.5	54.7	10
DeeperCut [11]	87.9	84.0	71.9	63.9	68.8	63.8	58.1	71.2	230
Ours	**93.7**	**91.4**	**81.4**	**72.5**	**77.7**	**73.0**	**68.1**	**79.7**	**0.005**
Full testing set									
DeeperCut [11]	78.4	72.5	60.2	51.0	57.2	52.0	45.4	59.5	485
Iqbal et al. [12]	58.4	53.9	44.5	35.0	42.2	36.7	31.1	43.1	10
Ours (one scale)	89.0	84.9	74.9	64.2	71.0	65.6	58.1	72.5	0.005
Ours	**91.2**	**87.6**	**77.7**	**66.8**	**75.4**	**68.9**	**61.7**	**75.6**	**0.005**

Figure 2.8: OpenPose Results on the MPII dataset (Simon et al., 2017)

Chapter 3: Multi-stream architecture for video classification

Unlike images which contained of static visual information of a single frames, video consist of more information by having more frames and temporal components like motion information and audio in each individual frames which is rarely exploited. The previous works have focused on low-level raw representations while our proposed method is the first to explore an intermediate representation that is more semantically rich. Figure 3.1 shows that the proposed architecture which is divided into 2 streams in order to recognize object and human pose as shown in the figure below. Each of the stream is constructed by the ResNet-50 which is a state-of-the-art CNN architecture to extract the static visual information and human posture information. Our system makes the final prediction by combining the Softmax class-scores of both streams through average fusion.

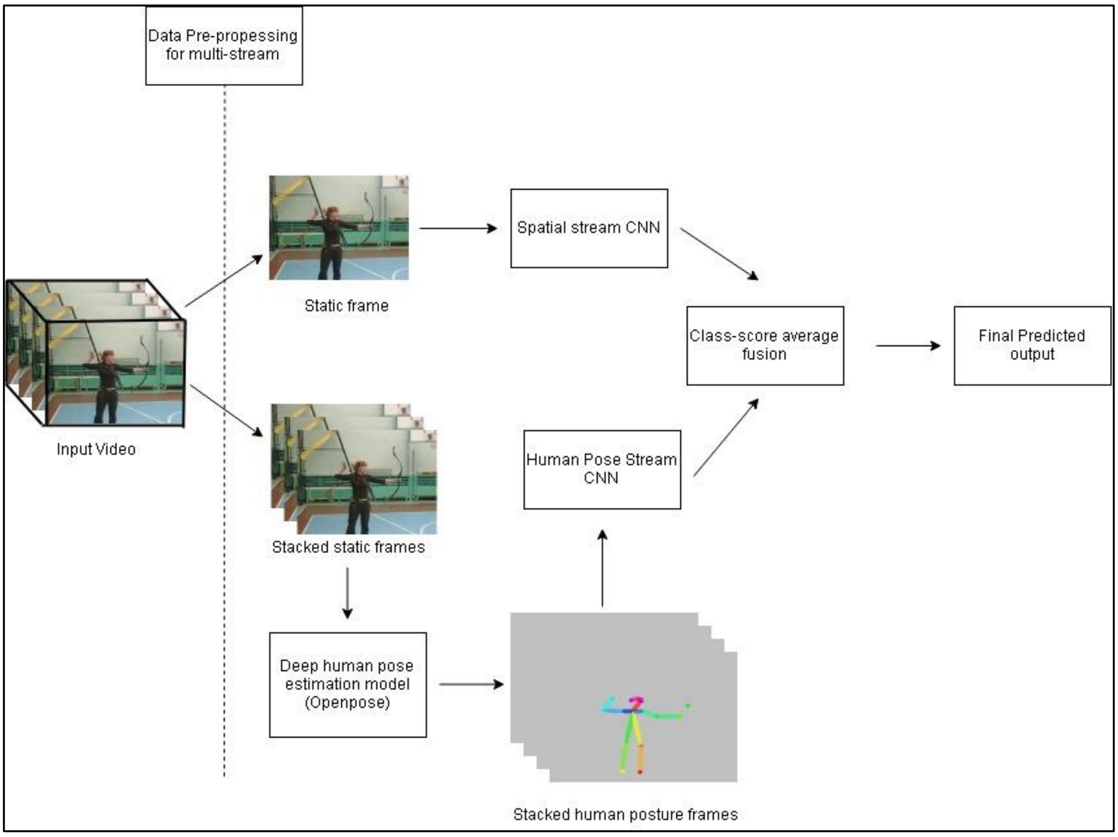

Data Pre-propessing for multi-stream
Input Video
Static frame
Spatial stream CNN
Class-score average fusion
Final Predicted output
Stacked static frames
Human Pose Stream CNN
Deep human pose estimation model (Openpose)
Stacked human posture frames

Figure 3.1: Overview of the full architecture

3,1 CNN configuration

The CNN architecture of our proposed system is built based on the residual network (ResNet-50) architecture as it is one of the best architecture available nowadays which having a good performance in term of its accuracy while maintaining a low amount of the parameters while having a deeper structure compared to the other state-of-the-art architectures as shown in the Figure 3.2.

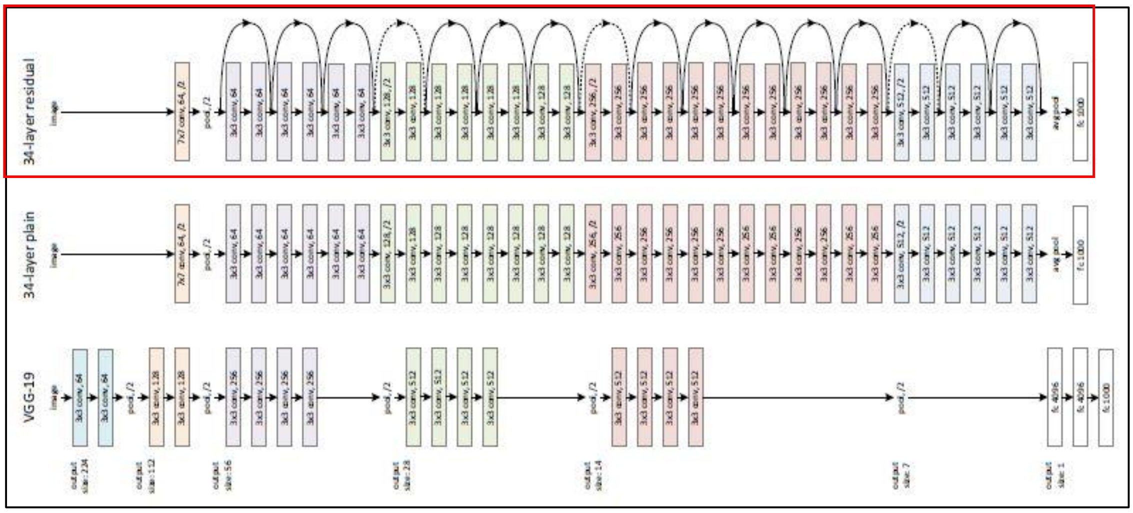


Figure 3.2: Comparison between the Residual Network and other architectures (He et al., 2015)

The graph of the ResNet-50 full architecture which plotted by the Keras framework is included in the Appendix for reference.

3.2 Spatial Stream CNN

The spatial information of the video can be obtained within each video frames. The spatial CNN works on individual still frames from the video by extracting and learning the features of the static appearance randomly sampled across the video as some frames are containing more information than the others. However, the performance of the spatial CNN is limited due to the limited amount of the video dataset available. In order to resolve this issue, pre-trained models that was trained on large-scale dataset namely the ImageNet (Russakovsky et al., 2015). This process is known as transfer learning, which has been shown to dramatically improve the performance of action classification.

Each stream is trained separately through transfer learning by applying the pre-trained ResNet-50 model that provided in the Keras library. However, application of the pre-trained are different due to the datasets of both streams:

- Spatial stream CNN
    - The pre-trained CNN in the spatial stream is treated as a fixed feature extractor as the dataset of these 26 classes are similar to some classes within the ImageNet dataset.
    - In our scenario, all the layers except the last fully connected layer of the ResNet-50 model are being set to non-trainable in order to avoid the weights of those layers being updated during the training process as most of the features learnt in the front layers are similar.
    - The last fully connected of the ResNet-50 is removed as it is the classifier which specifically trained to classify 1000 different classes within the ImageNet dataset and replaced by a new fully connected layer which acted as the classifier for 26 classes of dataset.

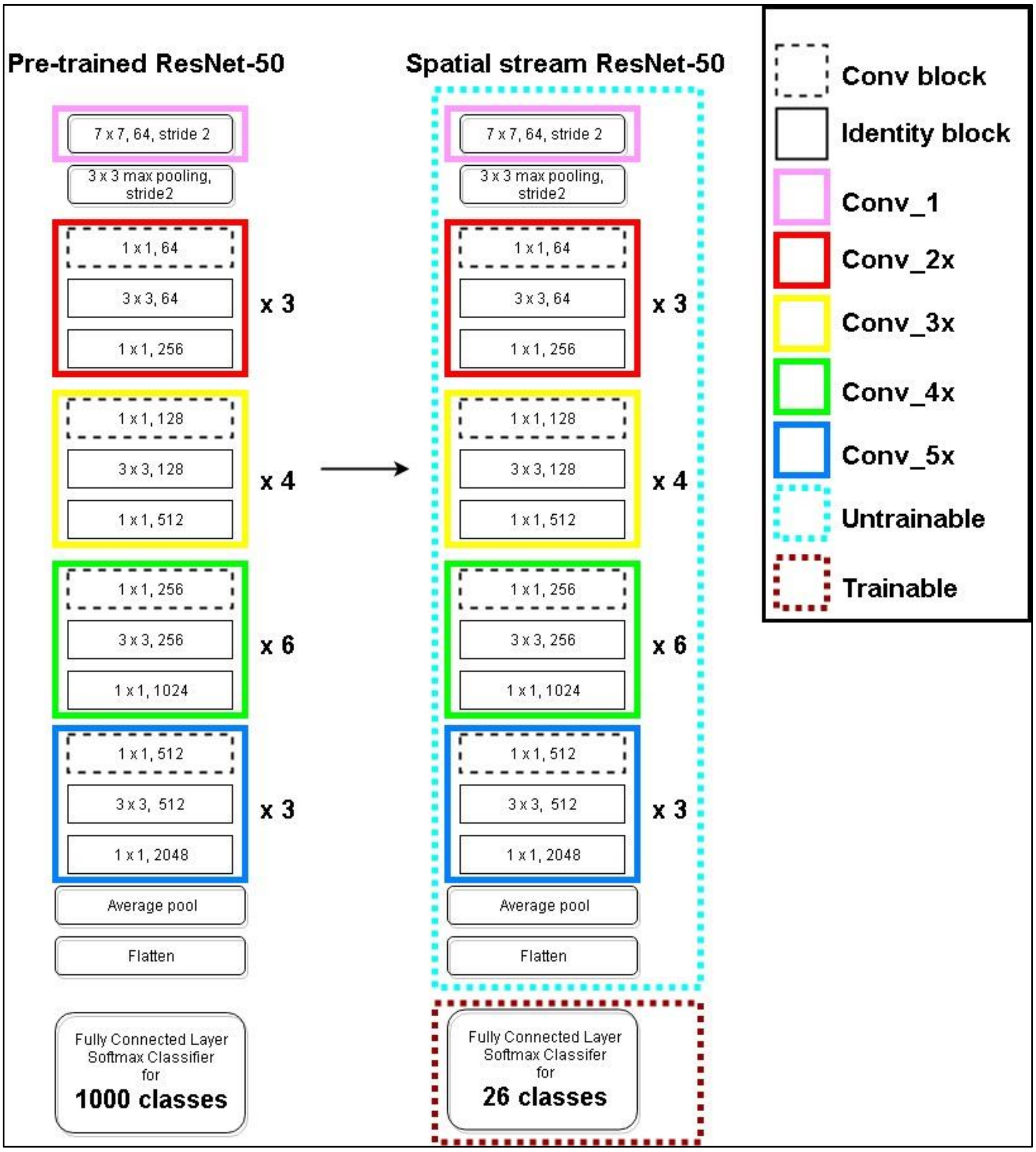


Figure 3.3: Transfer learning in spatial stream. The figure shows the ResNet-50 within the spatial stream which used the pre-trained model as a fixed features extractor as the transfer learning is performed between the same modality.

3.3 Human Pose Stream CNN

As the spatial CNN operates on raw video frames which contains both background and foreground information. However, for action classification are typically human actions which can be carried out in various environments like indoor or outdoor. However, although the background is different, each human action within the same category shared a same characteristic by having a similar series of postures which can be used to generalize the human action better during the recognition process. The human pose CNN of our proposed architecture are used to exploit the human posture within the video which takes the inputs of multiple stacked human pose frames with the background removed.

The human pose modality is selected as our proposed modality due to it is rarely being exploited as well as it able to reduce the background noises and concentrated more on the foreground information which is a set of the human posture. Furthermore, the pre-trained model of the OpenPose framework is also available in the GitHub which can be easily obtained.

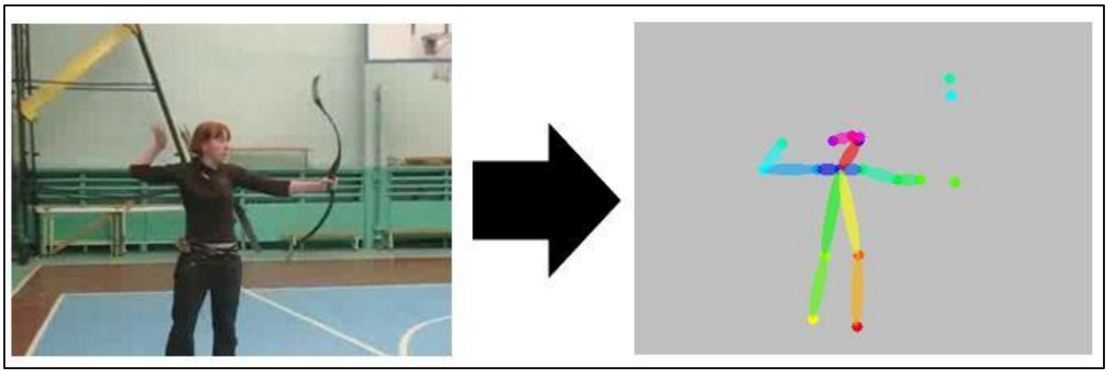


Figure 3.4: Video frame to Human pose frame

- Human pose stream CNN
    - The pre-trained CNN in the human pose stream is used as a weight initialization as the dataset (human pose frames) of the human pose stream is too vary from the ImageNet dataset (raw images).

- Furthermore, the input volume of the human pose stream (224 x 224 x 30) which is also different compared to the input volume (224 x 224 x 3) of the ImageNet pre-trained ResNet-50 model.
- As all the convolutional layer in the ResNet-50 are performing 2D-convolution which indicated that the depth of the convolutional filters need to match with the depth of the input as it only able to filter across the 2D space.
- All 64 filters (7, 7, 3) within the Conv_1 layer are duplicated 10 times and stacked together to form the filters of (7, 7, 30) in order to reuse the weight of the pre-trained model in the human pose stream CNN for initialization.
- The rest of the configuration are similar as the spatial stream except all the layers within the human pose stream are trainable.

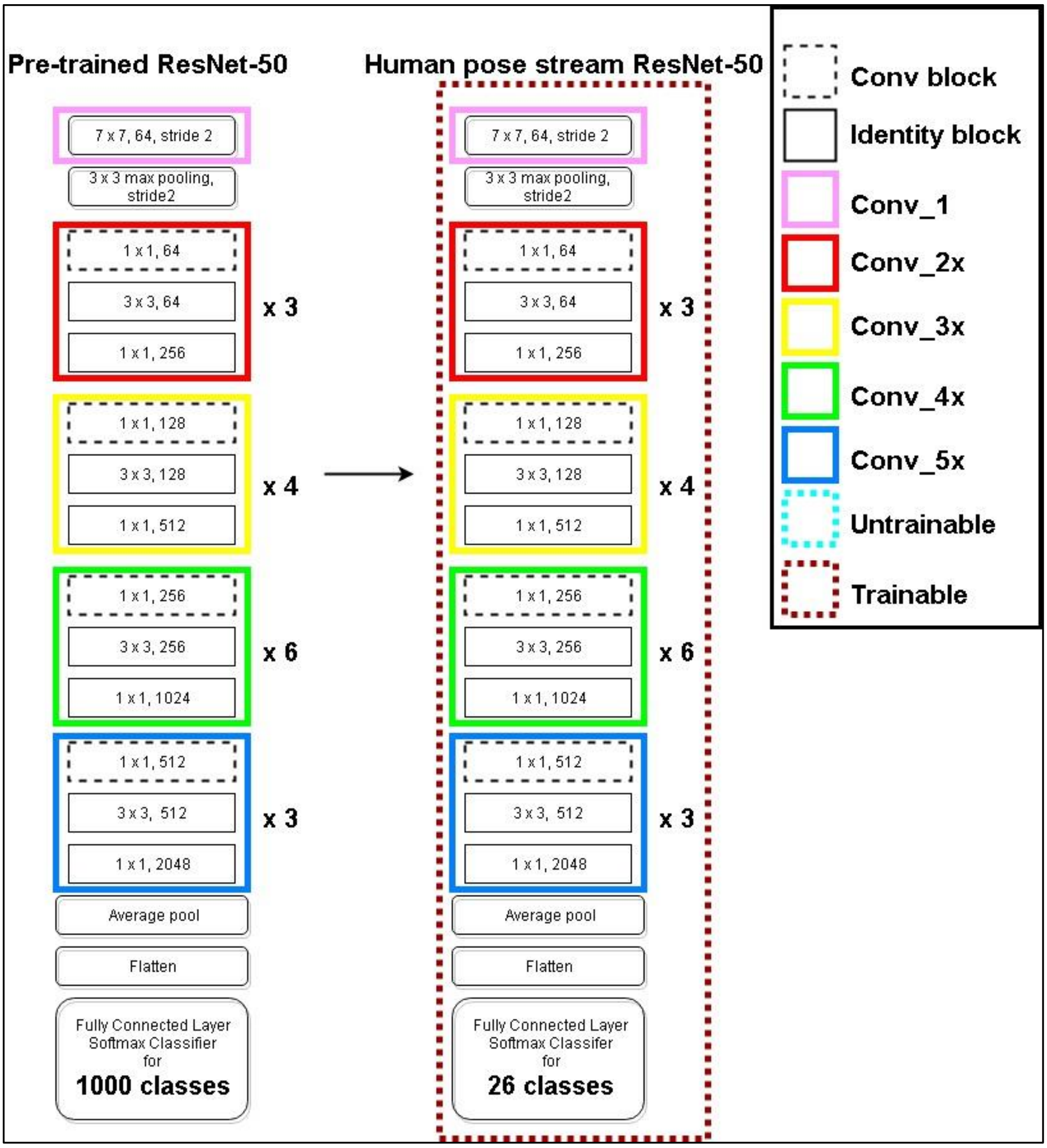


Figure 3.5: Transfer learning in human pose stream. The figure shows the ResNet-50 within the human pose stream which used the pre-trained model for weight initialization as the transfer learning is performed across the different modalities.

3.4 Fusing the 2 streams

Instead of implementing several new fully connected layers on top of the last fully connected layer to combine both streams, our proposed architecture applied average fusion across the Softmax class-scores of spatial and human pose stream to prevent overfitting issue. Figure 3.5 shows the overview of the proposed multi-stream network in details.

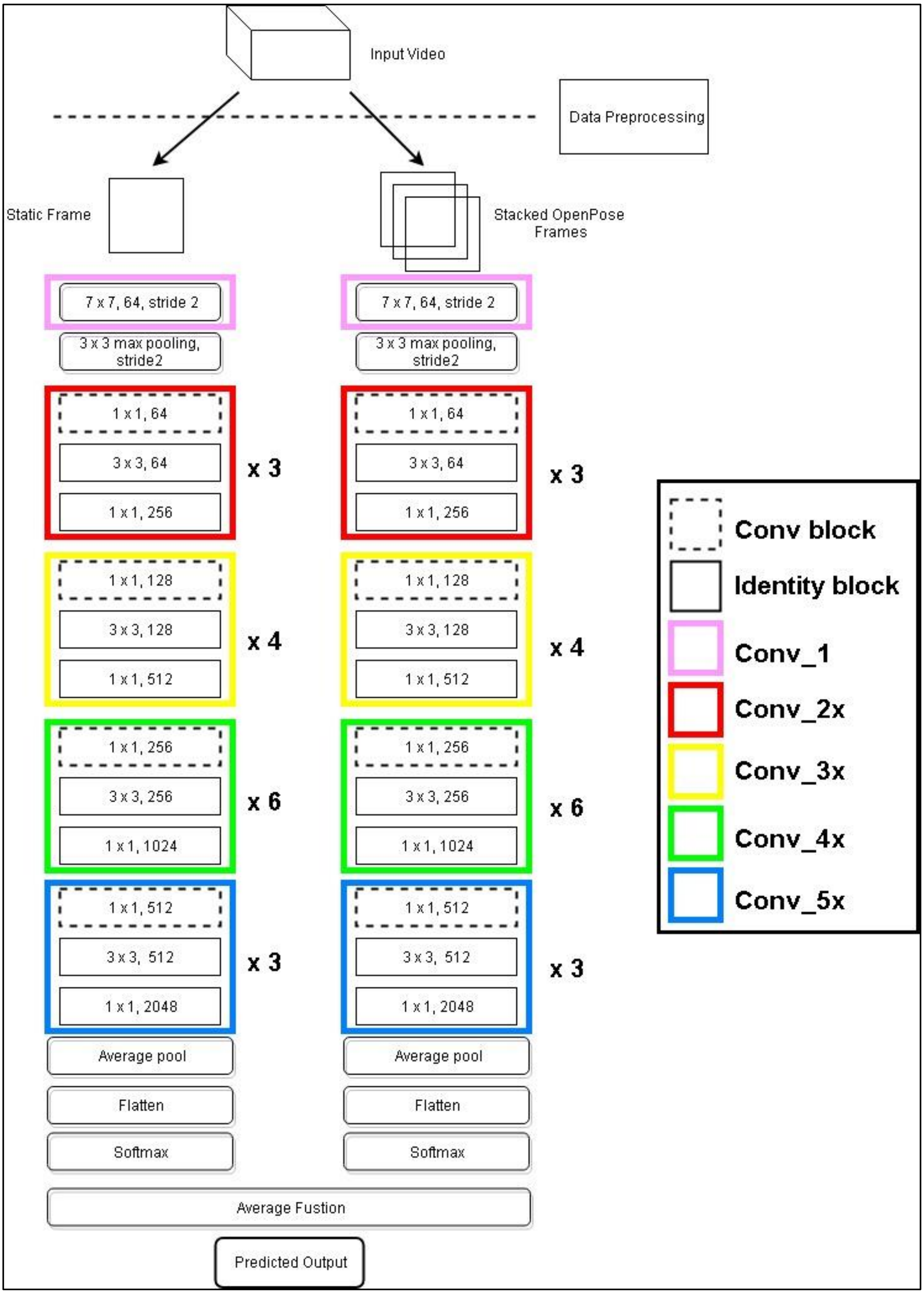


Figure 3.5: Overview of the proposed multi-stream network

Chapter 4: Implementation details of the proposed system

4.1 Overall design specification

- The system is implemented in Python language with the open source software library, OpenCV and Keras with Tensorflow backend which support the GPU's acceleration during the training process.
- The training process is accelerated by using a Nvidia GTX1080 GPU with 8GB of GDDR5X of video RAM.

4.2 Data Preprocessing and Data Augmentation

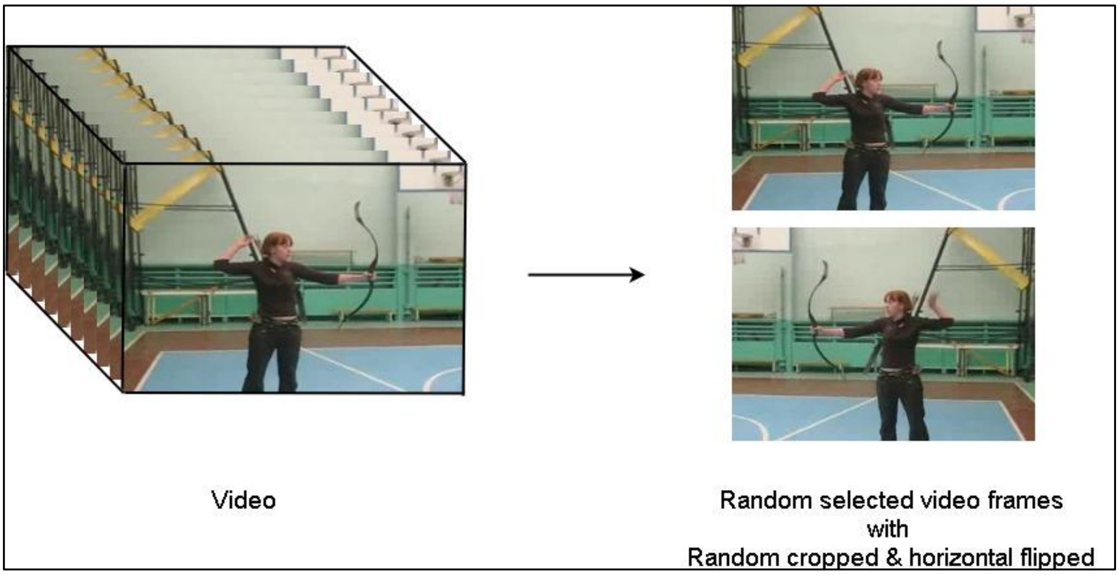

Video
Random selected video frames
with
Random cropped & horizontal flipped

Figure 4.1: Spatial stream input preprocessing

For the spatial stream, the input is a single image of resolution of 320 x 240 (width x height) is randomly selected from each input video. Then following the standard practice, we perform the data-preprocessing and data augmentation:

- Random cropping
- Random horizontal flipped
- Zero-centered data

The process consists of the random horizontal flipping and random cropping to form a resolution of 224 x 224 x 3 (width x height x RGB channels) to match the size of the

ImageNet pre-trained model's input. All the video frames are pre-generated before the training process to reduce the computational time.

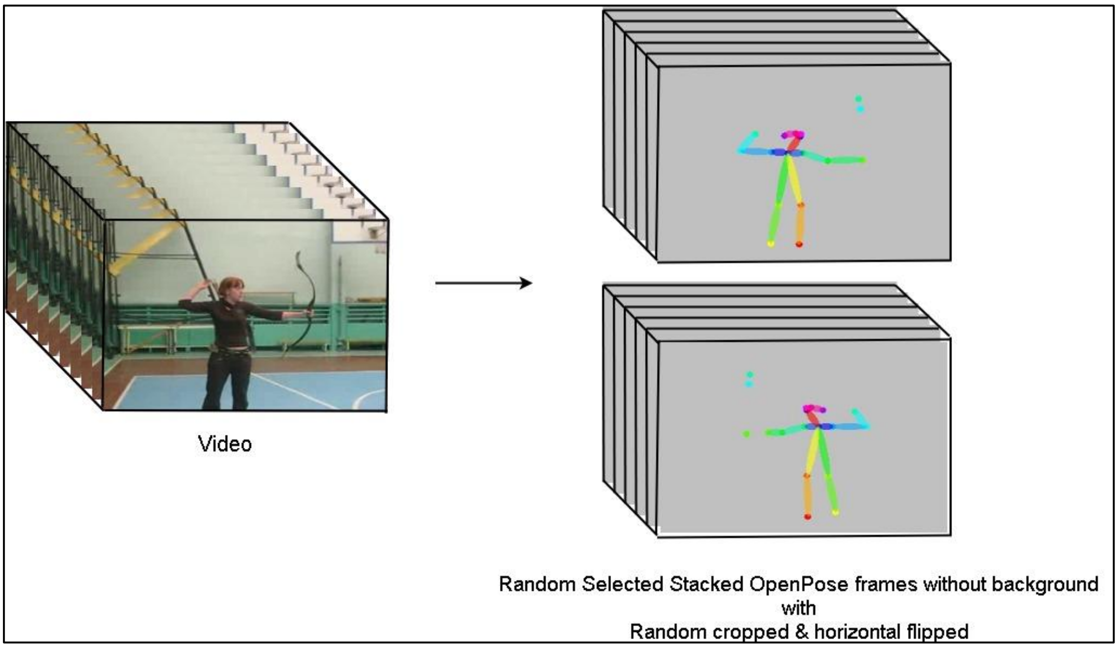

Video
Random Selected Stacked OpenPose frames without background
with
Random cropped & horizontal flipped

Figure 4.2: Human pose stream input preprocessing

For the human pose stream, given a video, the raw video frames used to generate the human pose frames through the generative model (OpenPose). 10 human pose frames are stacked together to form the input of the human pose stream. The data augmentation is same for both spatial and human pose stream.

4.3 Training and validation setup

Raw video and OpenPose frames are used as the input for the proposed architecture instead of whole video, where they are pre-generated before the training process. The weights of the network are updated by applied the Adam optimizer where the learning rate is set to 0.001. For each iteration, a batch of 2555 samples from the training set and 1011 samples from the testing set are selected for the training and validation process in both stream and undergoes the data preprocessing and augmentation which mentioned previously. Both streams are trained for 40k iterations separately and 4k iterations for the fusion scenario with the batch size of 32 on the spatial and 16 on the human pose stream and fusion scenario. The table below showed the training time for each stream.

Network	Training time (days)
Spatial CNN	3
Human pose CNN	5
2-stream CNN	1

Table 4.1: Training time for each stream

The size of the Human pose CNN’s input is larger which required longer training time due to a smaller batch size which is limited to capacity of the video RAM of the GPU while the 2 stream CNN is trained for only 4k iteration within 1 day due to the time constraint.

4.4.5 Testing setup

A constant amount of frames (25 for our case) are extracted from each input video with equally temporal spacing to ensure frames are being able to sample from the different part of the video equally. Each of them are further preprocessed by cropping from 4 corners and the center with and without the horizontal flipping to obtain extra 10 CNN inputs. The class score of each sampled frames are generated and the average of the samples are generated to compute the class score of the entire video. 10-cropped testing is performed for both streams and the 2-stream architecture.

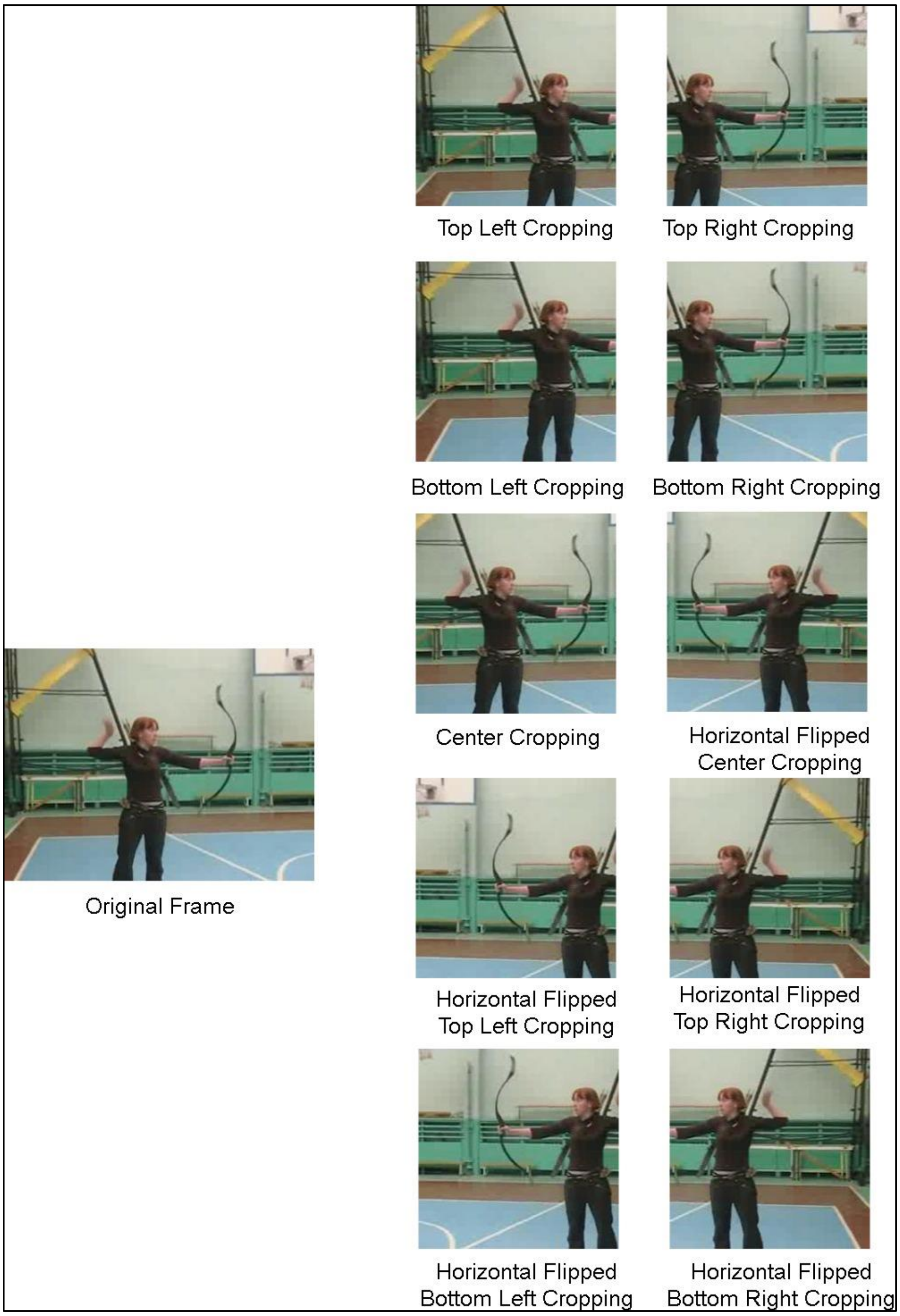

Top Left Cropping
Top Right Cropping
Bottom Left Cropping
Bottom Right Cropping
Center Cropping
Horizontal Flipped Center Cropping
Original Frame
Horizontal Flipped Top Left Cropping
Horizontal Flipped Top Right Cropping
Horizontal Flipped Bottom Left Cropping
Horizontal Flipped Bottom Right Cropping

Figure 4.3: 10-cropped testing samples

Chapter 5: Evaluation

5.1 Datasets used and evaluation protocol

The datasets which used for the training and evaluation is the UCF-101 (Soomro et al., 2012). It consists of around 13K videos divided into 101 categories. It divided into 3 splits each for both the training (9.5K) and testing (3.5K) sets. However, the results in this report is only based on 26 classes the first split of the training (2555 videos) and testing (1011 videos) set of the UCF-101 dataset due to the time constraints. The evaluated classes are:

- Archery
- Baby Crawling
- Balance Beam
- Band Marching
- Baseball Pitch
- Basketball
- Basketball Dunk
- Bench Press
- Biking
- Billiards
- Boxing Punching Bag
- Boxing Speed Bag
- Ice Dancing
- Juggling Balls
- Jumping Jack
- Jump Rope
- Punch
- Tai Chi
- Tennis Swing
- Throw Discus
- Trampoline Jumping
- Volleyball Spiking
- Walking with Dog
- Wall Pushups
- Writing On Board
- Yo-Yo

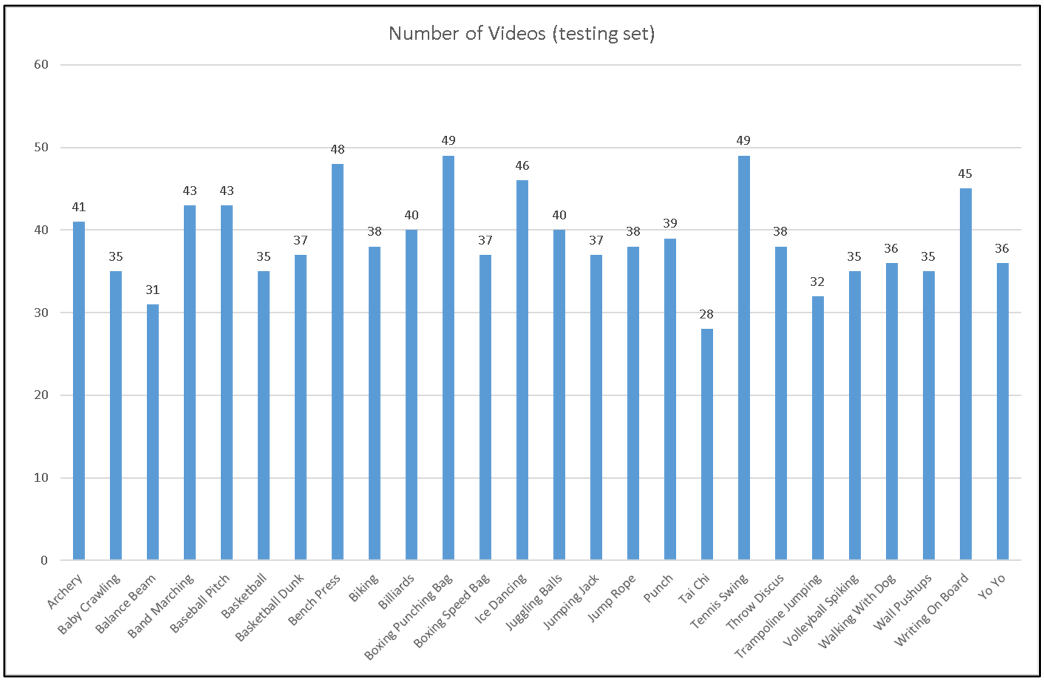
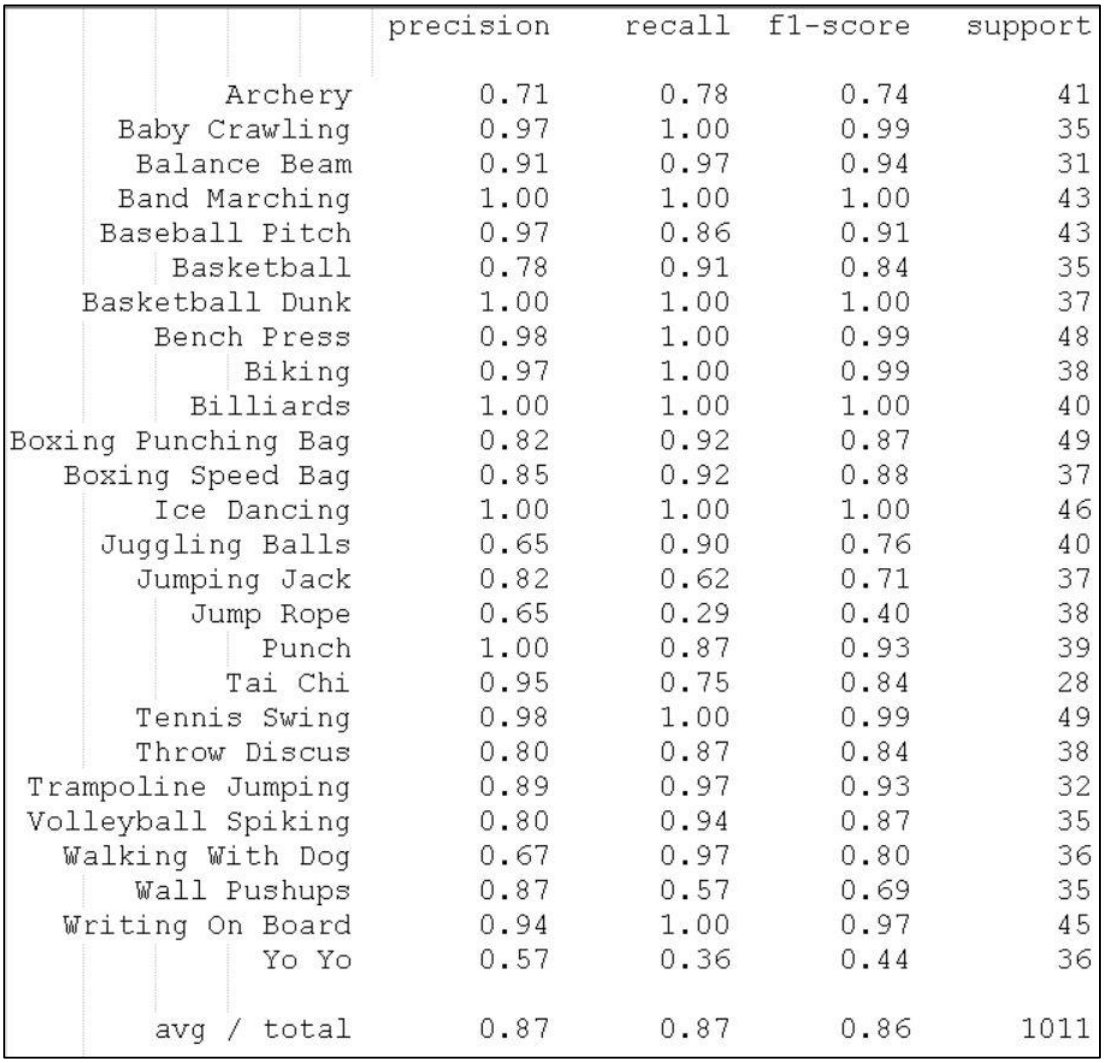


Figure 5.1: Number of videos in testing set

5.2 Discussion of the result

Architecture	Accuracy
Spatial CNN	87.1%
Human pose CNN	71.0%
2 stream CNN	91.3%

Table 5.1: 10-cropped testing results

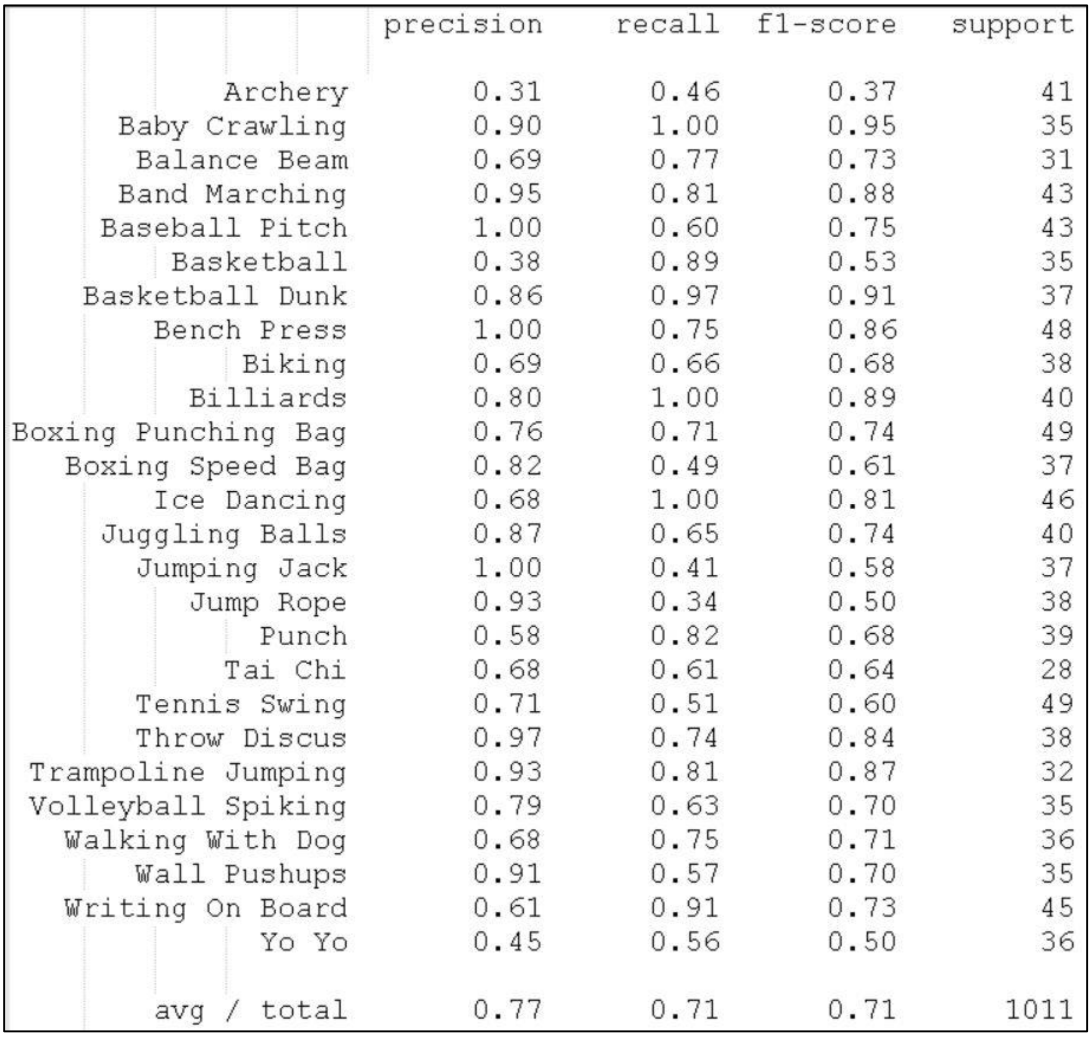


```
                    precision    recall  f1-score   support

           Archery       0.71      0.78      0.74        41
     Baby Crawling       0.97      1.00      0.99        35
      Balance Beam       0.91      0.97      0.94        31
     Band Marching       1.00      1.00      1.00        43
    Baseball Pitch       0.97      0.86      0.91        43
        Basketball       0.78      0.91      0.84        35
   Basketball Dunk       1.00      1.00      1.00        37
       Bench Press       0.98      1.00      0.99        48
            Biking       0.97      1.00      0.99        38
         Billiards       1.00      1.00      1.00        40
Boxing Punching Bag       0.82      0.92      0.87        49
   Boxing Speed Bag       0.85      0.92      0.88        37
       Ice Dancing       1.00      1.00      1.00        46
    Juggling Balls       0.65      0.90      0.76        40
      Jumping Jack       0.82      0.62      0.71        37
         Jump Rope       0.65      0.29      0.40        38
             Punch       1.00      0.87      0.93        39
           Tai Chi       0.95      0.75      0.84        28
      Tennis Swing       0.98      1.00      0.99        49
      Throw Discus       0.80      0.87      0.84        38
 Trampoline Jumping       0.89      0.97      0.93        32
 Volleyball Spiking       0.80      0.94      0.87        35
   Walking With Dog       0.67      0.97      0.80        36
      Wall Pushups       0.87      0.57      0.69        35
   Writing On Board       0.94      1.00      0.97        45
             Yo Yo       0.57      0.36      0.44        36

       avg / total       0.87      0.87      0.86      1011
```

Figure 5.2: Classification report for spatial stream

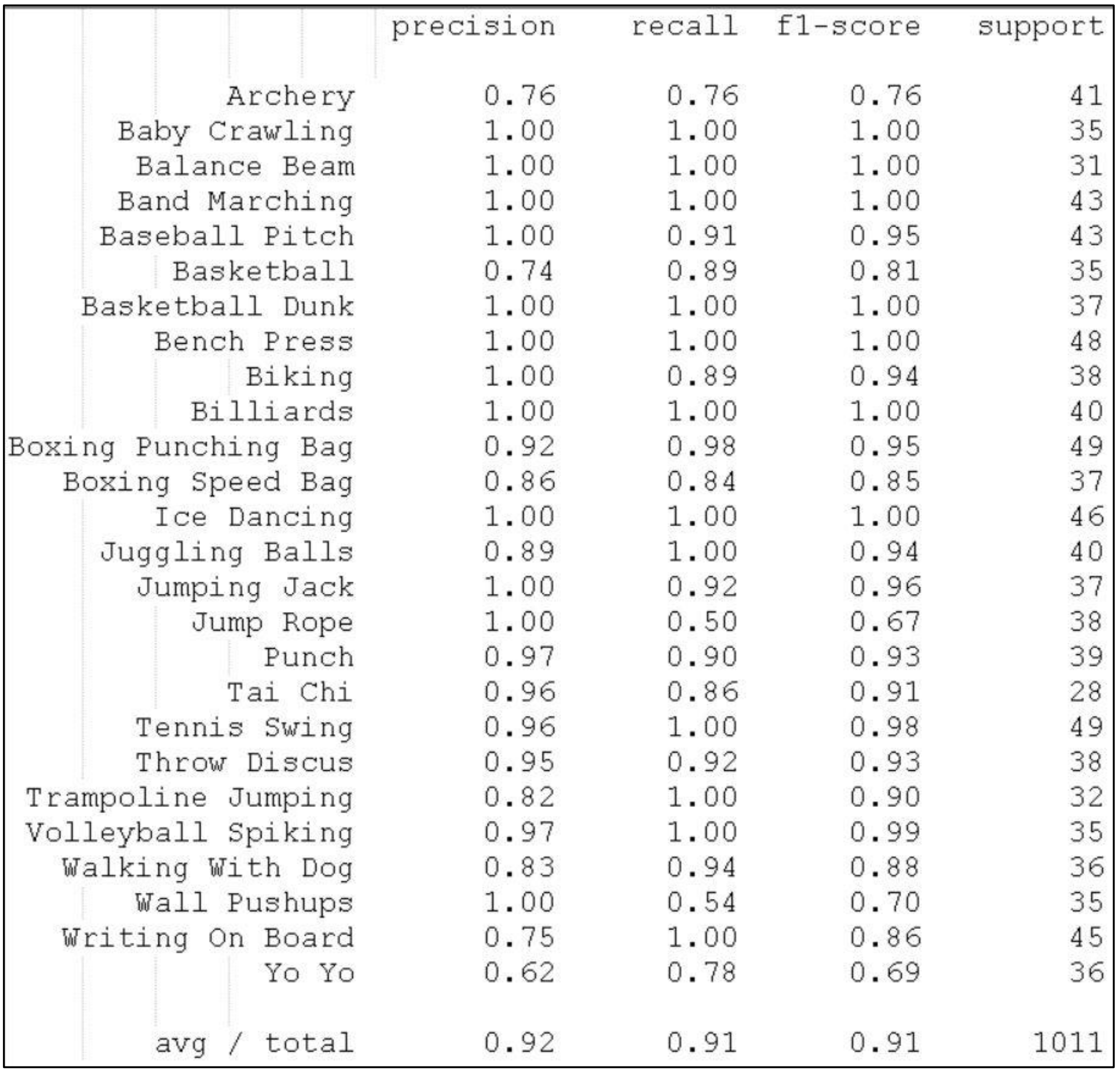


```
                     precision    recall  f1-score   support

            Archery       0.31      0.46      0.37        41
      Baby Crawling       0.90      1.00      0.95        35
       Balance Beam       0.69      0.77      0.73        31
      Band Marching       0.95      0.81      0.88        43
     Baseball Pitch       1.00      0.60      0.75        43
         Basketball       0.38      0.89      0.53        35
    Basketball Dunk       0.86      0.97      0.91        37
       Bench Press        1.00      0.75      0.86        48
             Biking       0.69      0.66      0.68        38
          Billiards       0.80      1.00      0.89        40
Boxing Punching Bag       0.76      0.71      0.74        49
   Boxing Speed Bag       0.82      0.49      0.61        37
        Ice Dancing       0.68      1.00      0.81        46
     Juggling Balls       0.87      0.65      0.74        40
       Jumping Jack       1.00      0.41      0.58        37
          Jump Rope       0.93      0.34      0.50        38
              Punch       0.58      0.82      0.68        39
            Tai Chi       0.68      0.61      0.64        28
       Tennis Swing       0.71      0.51      0.60        49
       Throw Discus       0.97      0.74      0.84        38
 Trampoline Jumping       0.93      0.81      0.87        32
 Volleyball Spiking       0.79      0.63      0.70        35
   Walking With Dog       0.68      0.75      0.71        36
       Wall Pushups       0.91      0.57      0.70        35
   Writing On Board       0.61      0.91      0.73        45
              Yo Yo       0.45      0.56      0.50        36

        avg / total       0.77      0.71      0.71      1011
```

Figure 5.3: Classification report for human pose stream



```
                      precision    recall  f1-score   support

            Archery        0.76      0.76      0.76        41
      Baby Crawling        1.00      1.00      1.00        35
       Balance Beam        1.00      1.00      1.00        31
      Band Marching        1.00      1.00      1.00        43
     Baseball Pitch        1.00      0.91      0.95        43
         Basketball        0.74      0.89      0.81        35
    Basketball Dunk        1.00      1.00      1.00        37
        Bench Press        1.00      1.00      1.00        48
             Biking        1.00      0.89      0.94        38
          Billiards        1.00      1.00      1.00        40
Boxing Punching Bag        0.92      0.98      0.95        49
   Boxing Speed Bag        0.86      0.84      0.85        37
        Ice Dancing        1.00      1.00      1.00        46
     Juggling Balls        0.89      1.00      0.94        40
       Jumping Jack        1.00      0.92      0.96        37
          Jump Rope        1.00      0.50      0.67        38
              Punch        0.97      0.90      0.93        39
            Tai Chi        0.96      0.86      0.91        28
       Tennis Swing        0.96      1.00      0.98        49
       Throw Discus        0.95      0.92      0.93        38
 Trampoline Jumping        0.82      1.00      0.90        32
 Volleyball Spiking        0.97      1.00      0.99        35
   Walking With Dog        0.83      0.94      0.88        36
       Wall Pushups        1.00      0.54      0.70        35
   Writing On Board        0.75      1.00      0.86        45
              Yo Yo        0.62      0.78      0.69        36

        avg / total        0.92      0.91      0.91      1011
```

Figure 5.4: Classification report for 2 stream CNN

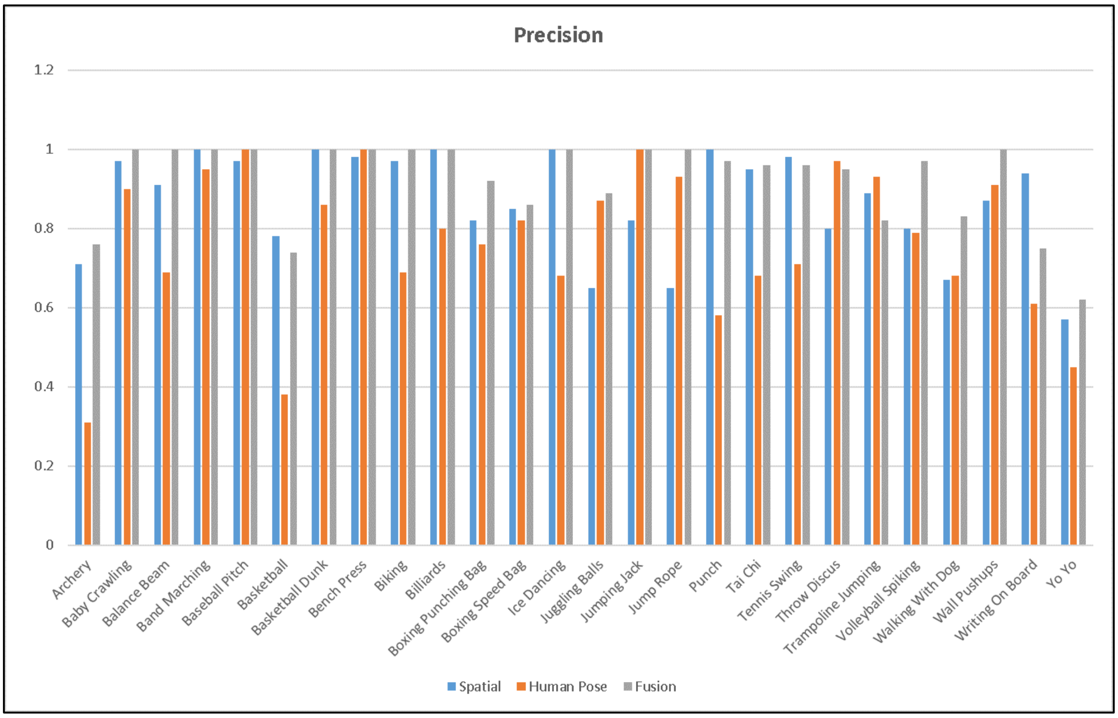


Figure 5.5: Precision of spatial, human pose and 2 stream

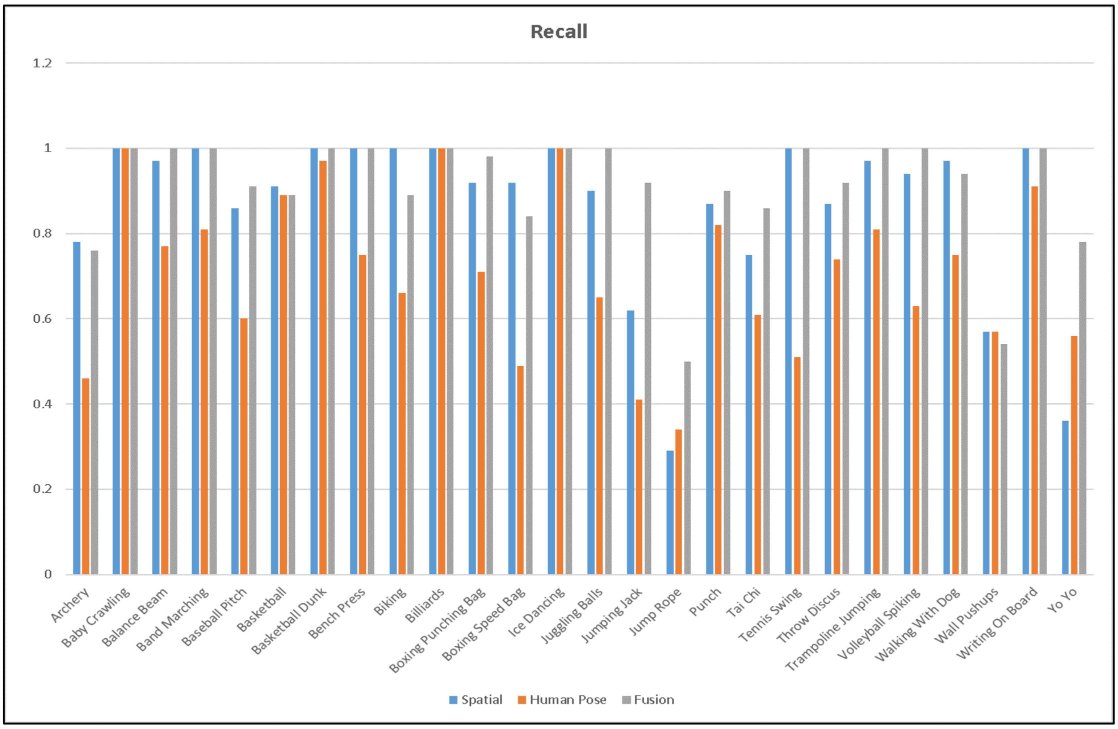


Figure 5.6: Recall of spatial, human pose and 2 stream

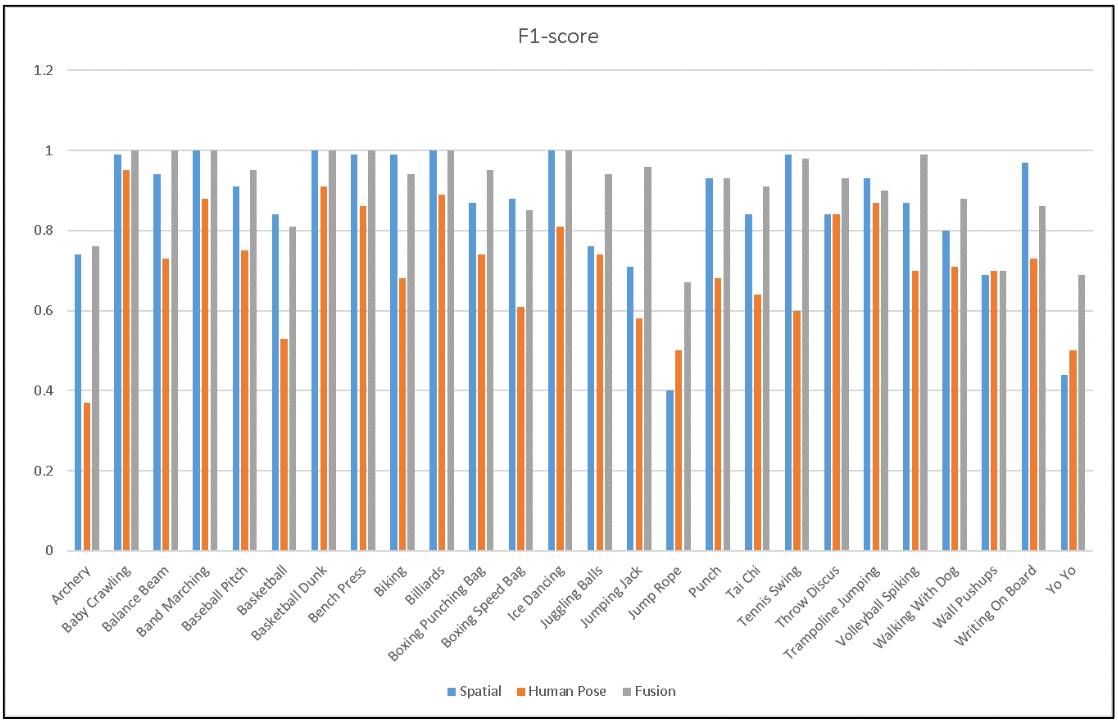


Figure 5.7: F1-score of spatial, human pose and 2 stream

From the Table 5.1, we can observe that the 2 stream CNN out-performs the spatial and human pose stream by considering multiple modalities within the video. It also showed that the potential of human pose stream in the video classification domain. However, based the Figure 5.5, 5.6, 5.7, we can observe that the human pose only helps to improve the performance of certain classes in the 2 stream architecture. Classes such as the Archery, Baby Crawling, Balance Beam, Baseball Pitch, Bench Press, Boxing Punching Bag, Juggling Balls, Jumping Jack, Jump Rope, Tai Chi, Throw Discus, Volleyball Spiking, Walking With Dog, Wall Pushups and Yo-Yo (15 classes) are having a clearer human pose due to human within the video is are clear enough to be detected which ended up with a better quality of human pose frames.

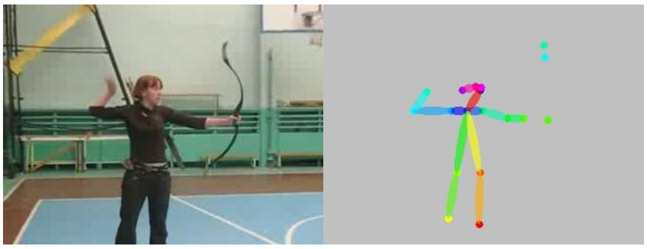


Figure 5.8: Raw frame and human pose frame for Archery action (g01_c02)

Classes such as Band Marching, Basketball Dunk, Billiards, Ice Dancing and Punch (5 classes) are remain unchanged compared to the performance of the spatial stream and 2 stream architecture due to the spatial stream is able to extract enough information from the raw frames for classification without the support of the human pose stream.

For classes such as Basketball, Biking, Boxing Speed Bag, Tennis Swing, Trampoline Jumping and Writing On Board, the human pose stream has worsened the performance of the 2 stream CNN as most of the classes have a human position or posture which is harder to be identity and detected and caused the human pose frames turn into the noises that distract the performance of the network.

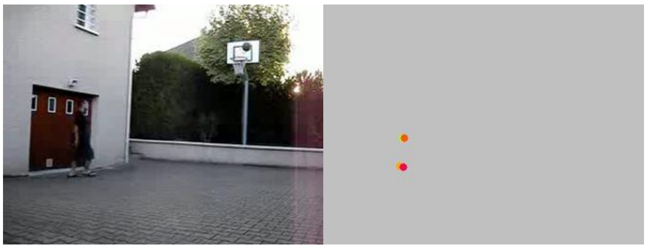


Figure 5.9: Raw frame and human pose frame for Archery action (g03_c06)

5.3 Transfer learning with pre-trained ResNet-50 model in CNNs

5.3.1 Spatial CNN

The performance of the spatial CNN is evaluated by using 2 scenarios:

i. Training from scratch on the UCF-101 dataset
ii. Using the pre-trained model as a fixed feature extractor on top of a new Softmax classifier.

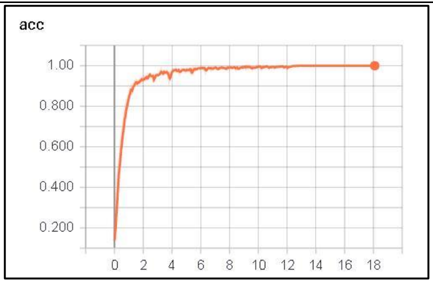
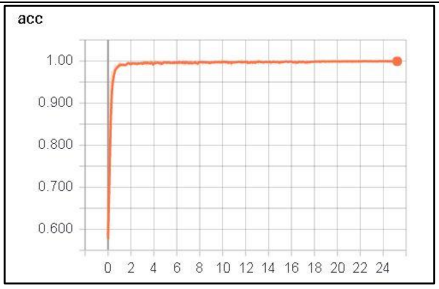
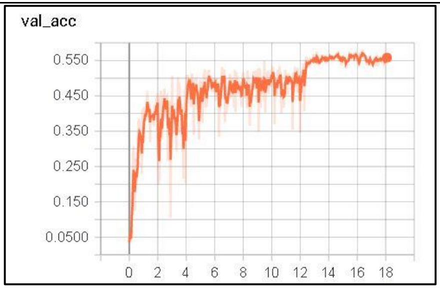
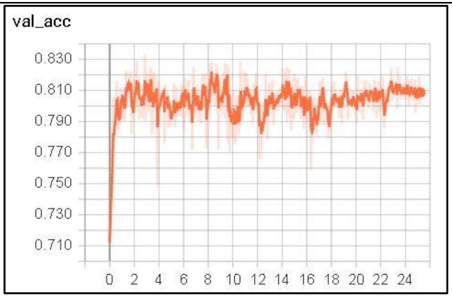
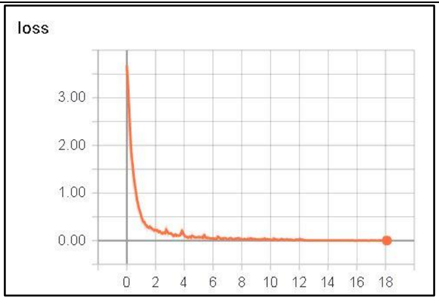
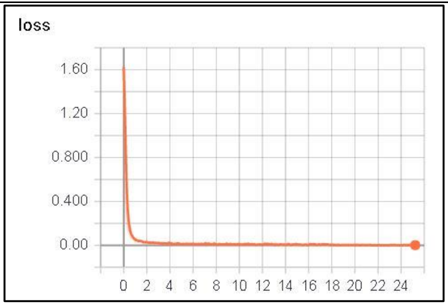
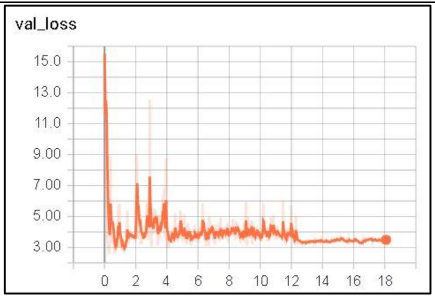
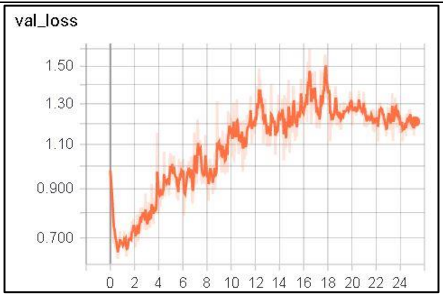
Type of result / Type of setting	i. Trained from scratch	ii. Pre-trained model as fixed feature extractor
Training accuracy		
Validation accuracy		
Training Loss		
Validation loss		

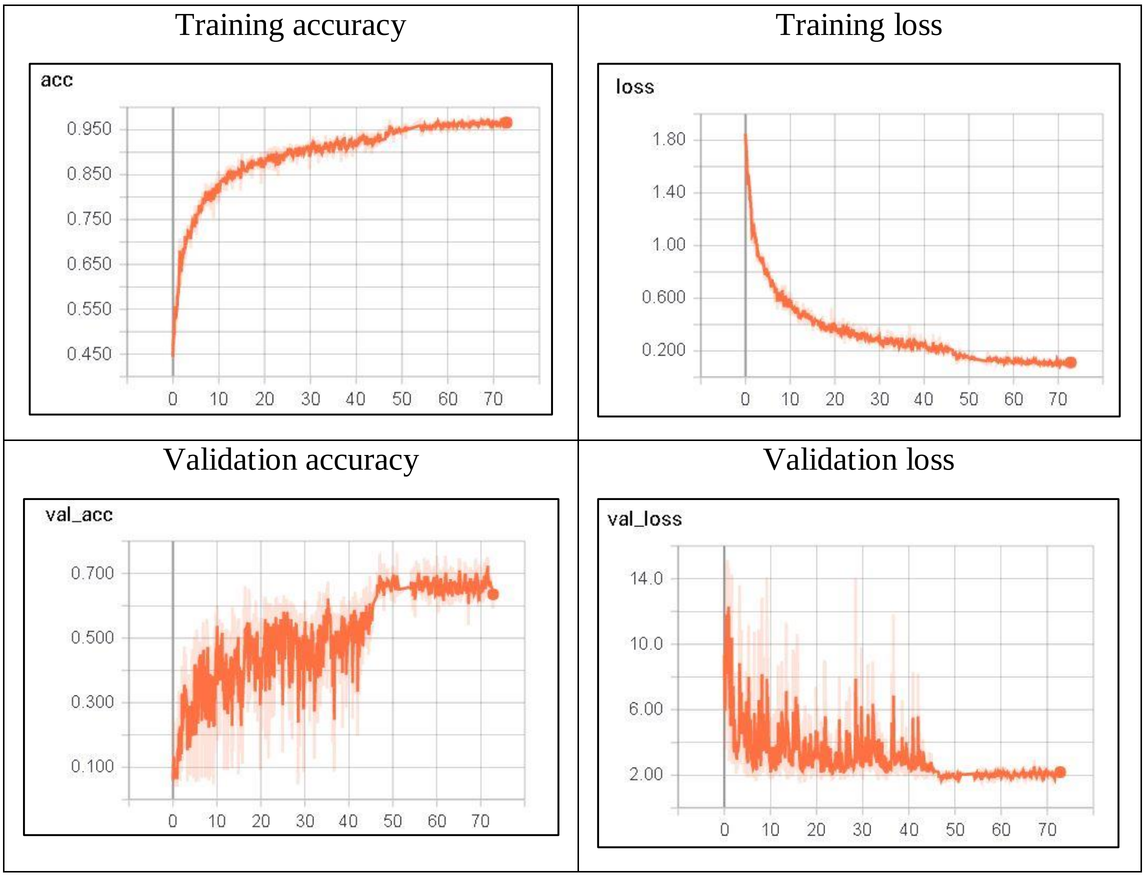
Figure 5.3.1: Trained from scratch VS Pre-trained model

Figure 5.2 shows the accuracy and loss of the training and validation across epochs. We can observe that the training accuracy of the spatial CNN with pre-trained model is converges compared to the spatial CNN trained from.

For the validation accuracy and loss, the spatial CNN with pre-trained out-performs the model trained from scratch where it delivers a higher accuracy and lower loss. The loss increases in the intermediate stage which shows that it is advisable to apply early stopping.

Furthermore, we can deduce that model that trained from scratch suffers from overfitting due to a high training accuracy and low validation accuracy. Using an appropriate pre-trained model is an important factor that enables model to learn faster and generalize better.

5.3.2 Human pose CNN



Training accuracy

Training loss

Validation accuracy

Validation loss

Figure 5.3.2: Human pose CNN training, validation loss and accuracy

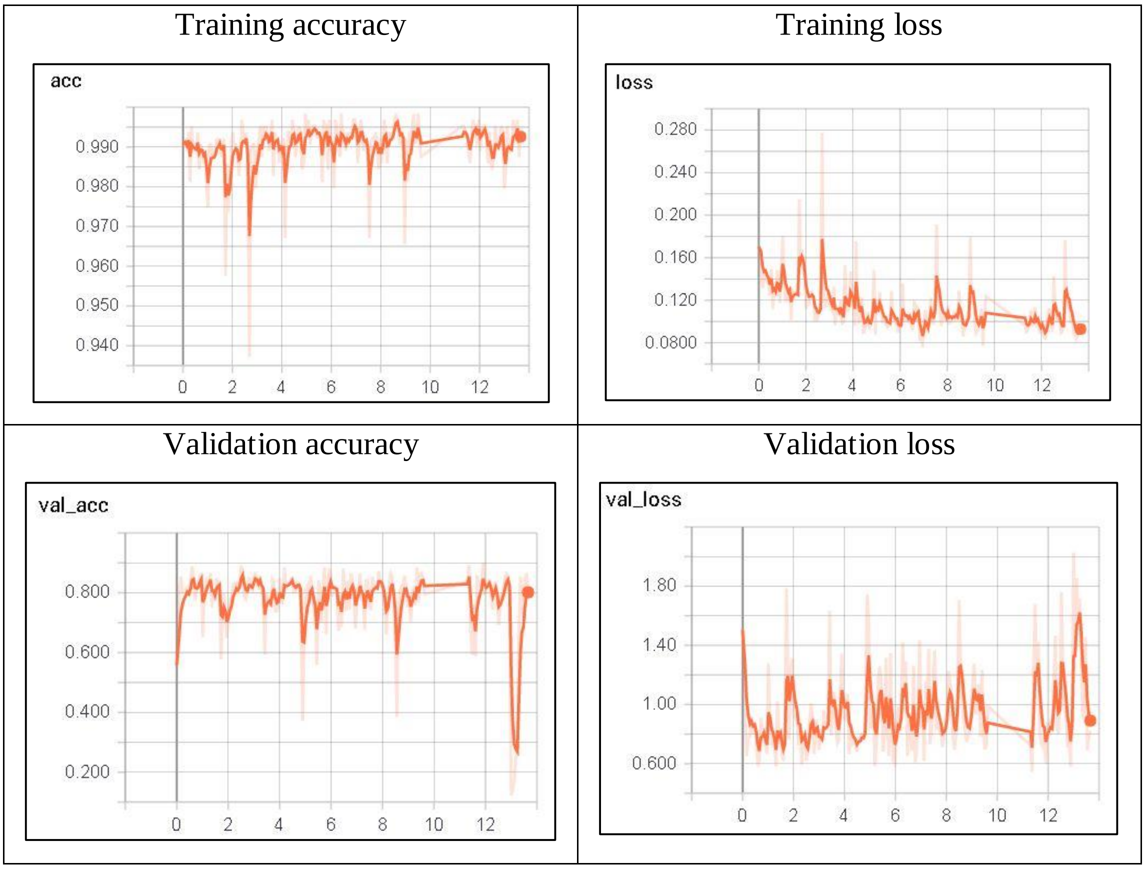
Figure 5.4 shows the that both the training accuracy and loss of human pose CNN converge slower compared to the spatial. This is due to the following reasons:

- All the layers in the human pose CNN are set to be trainable

  - Unlike the spatial CNN which only the last fully connected layer is trainable, the human pose CNN has more weights and parameters that required to updated when result in slower accuracy and loss converging
- Differences between the ImageNet datasets and human pose frames
  - As the 2 datasets are very different from each other, the human pose CNN requires more training time to learn the weights based on the human pose frames as the weights are initialized by the ImageNet datasets.

From Figure 5.3, we can observe that it is still outperformed the spatial CNN which trained from scratch which shows that the cross-modality pre-training is beneficial for our proposed human pose stream.

5.4 Average fusion of 2-stream CNN



Training accuracy

Training loss

Validation accuracy

Validation loss

Figure 5.3.3: 2 stream CNN training, validation loss and accuracy

From the figure above, we can observe that 2 stream CNN aslo required a longer training as it has even more the weights to learn compared to the spatial or human pose CNN.

Chapter 6: Conclusion

6.1 Discussion

Our proposed multi-stream architecture with the spatial stream combined with a new modality of the human pose by average fusion able to achieve better result than the spatial stream alone while the human pose stream alone is not beneficial due to the loss of the spatial information is critical. However, when both of the static visual information and a series of human pose information are considered together, it able to performed better classification than either stream alone which proved that the importance of the multi-modalities processing within the video.

This paper has showed that the cross-modality pre-training on the human pose stream with spatial stream weights able to alleviate the overfitting issue by using the weights of the spatial model as the weights to initialize the human pose stream.

6.2 Future works

As this is a pilot study which trained and evaluated on the partial UCF-101 dataset which consists of only 26 classes, full datasets with multiple splits can be considered which consists of 101 classes and 3 splits.

The performance of the proposed human pose stream can be further analyzed on different datasets such as the HMDB-51 datasets which consists of 51 classes.

The proposed human pose stream can be further extended by computing the optical flow on consecutive human pose frames which fully eliminated the background motion and only focused on the motion information of the foreground which generated when human carried out the action.

Modification of the proposed architecture can also be considered such as applying the LSTM RNN for longer motion information processing as the CNN can only handle the short-term motion information which can further boost the performance in term of the accuracy.

Bibliography

B. Zhang, L. Wang, Z. Wang, Y. Qiao and H. Wang, Real-time Action Recognition with Enhanced Motion Vector CNNs, in *CVPR, 2016.*

Baccouche, M., Mamalet, F., Wolf, C., Garcia, C. and Baskurt, A., 2011, November. Sequential deep learning for human action recognition. In *International Workshop on Human Behavior Understanding* (pp. 29-39). Springer Berlin Heidelberg.

Chollet, F and others, 2015. Keras Github. Available at: *github* https://github.com/keras-team/keras.

He, K., Zhang, X., Ren, S. and Sun, J., 2016. Deep Residual Learning for Image Recognition. IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016.

K. Soomro, A. R. Zamir, and M. Shah. UCF101: A dataset of 101 human actions classes form videos in the wild. CoRR, 2012

Karpathy, A., Toderici, G., Shetty, S., Leung, T., Sukthankar, R. and Fei-Fei, L., 2014. Large-scale video classification with convolutional neural networks. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition* (pp. 1725-1732).

L. Wang, Y. Xiong, Z. Wang, Y. Qiao, D. Lin, X. Tang and L. V. Gool, Temporal Segment Networks: Towards Good Practices for Deep Action Recognition, in *ECCV, 2016.*

Russakovsky, O., Deng, J., Su, H., Krause, J. Statheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. and Fei-Fei, L., 2015. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision, IJCV 2015.*

Bibliography

Simonyan, K. and Zisserman, A., 2014. Two-stream convolutional networks for action recognition in videos. In *Advances in Neural Information Processing Systems* (pp. 568-576).

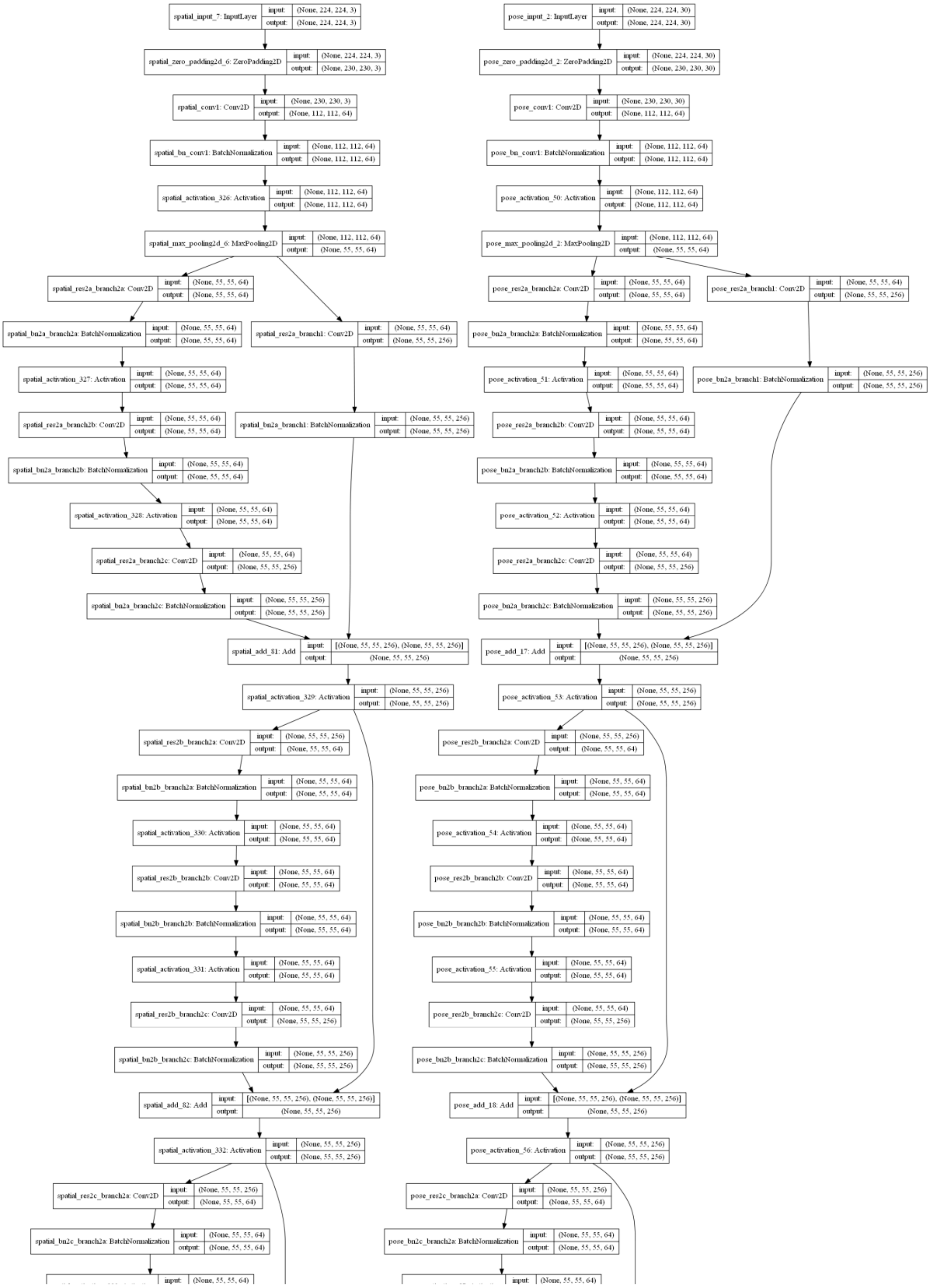
Simonyan, K. and Zisserman, A., 2014. Very Deep Convolutional Networks for Large-Scale Image Recognition. In *Proc. International Conference on Learning Representations.*

Z. Cao, T. Simon, S. Wei and Y. Sheikh, Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields, in *CVPR, 2017.*

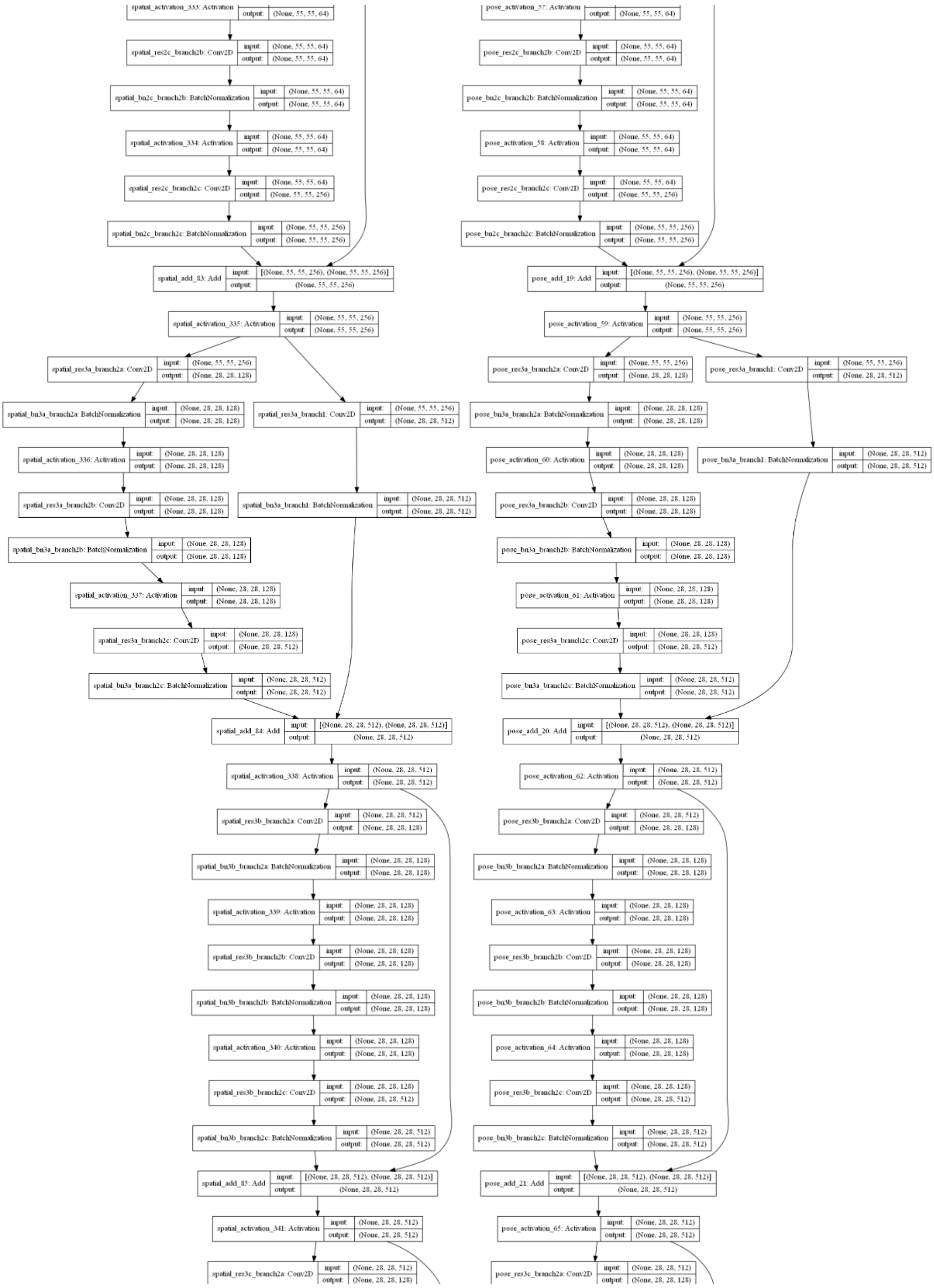
Z. Wu, Y.-G. Jiang, X. Wang, H. Ye and X. Xue, Multi-Stream Multi-Class Fusion of Deep Networks for Video Classification, in *ACM Multimedia, 2016.*

Appendix

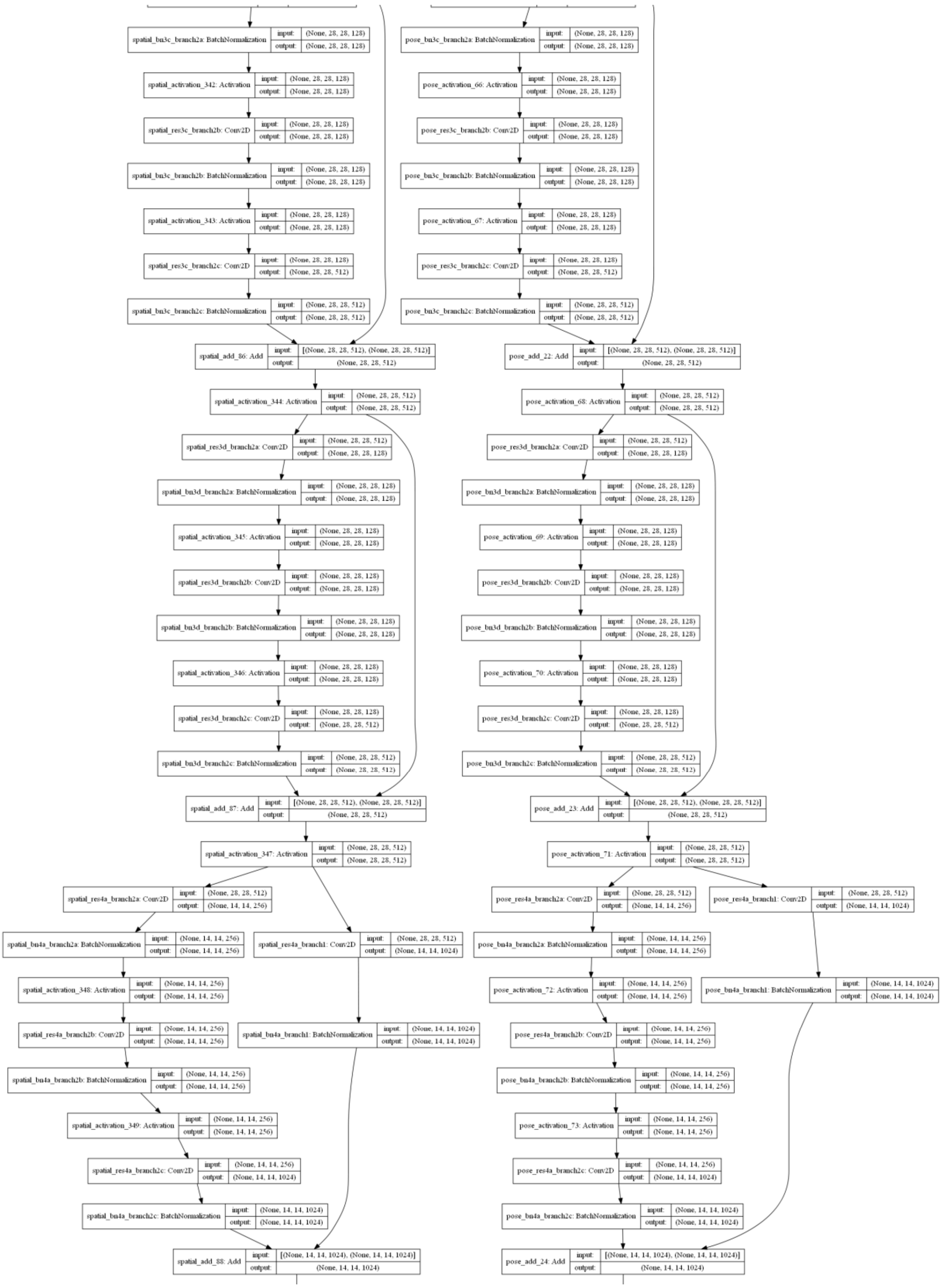
Appendix


spatial_input_7: InputLayer
input: (None, 224, 224, 3)
output: (None, 224, 224, 3)
spatial_zero_padding2d_6: ZeroPadding2D
input: (None, 224, 224, 3)
output: (None, 230, 230, 3)
spatial_conv1: Conv2D
input: (None, 230, 230, 3)
output: (None, 112, 112, 64)
spatial_bn_conv1: BatchNormalization
input: (None, 112, 112, 64)
output: (None, 112, 112, 64)
spatial_activation_326: Activation
input: (None, 112, 112, 64)
output: (None, 112, 112, 64)
spatial_max_pooling2d_6: MaxPooling2D
input: (None, 112, 112, 64)
output: (None, 55, 55, 64)
spatial_res2a_branch2a: Conv2D
input: (None, 55, 55, 64)
output: (None, 55, 55, 64)
spatial_bn2a_branch2a: BatchNormalization
input: (None, 55, 55, 64)
output: (None, 55, 55, 64)
spatial_res2a_branch1: Conv2D
input: (None, 55, 55, 64)
output: (None, 55, 55, 256)
spatial_activation_327: Activation
input: (None, 55, 55, 64)
output: (None, 55, 55, 64)
spatial_res2a_branch2b: Conv2D
input: (None, 55, 55, 64)
output: (None, 55, 55, 64)
spatial_bn2a_branch1: BatchNormalization
input: (None, 55, 55, 256)
output: (None, 55, 55, 256)
spatial_bn2a_branch2b: BatchNormalization
input: (None, 55, 55, 64)
output: (None, 55, 55, 64)
spatial_activation_328: Activation
input: (None, 55, 55, 64)
output: (None, 55, 55, 64)
spatial_res2a_branch2c: Conv2D
input: (None, 55, 55, 64)
output: (None, 55, 55, 256)
spatial_bn2a_branch2c: BatchNormalization
input: (None, 55, 55, 256)
output: (None, 55, 55, 256)
spatial_add_81: Add
input: [(None, 55, 55, 256), (None, 55, 55, 256)]
output: (None, 55, 55, 256)
spatial_activation_329: Activation
input: (None, 55, 55, 256)
output: (None, 55, 55, 256)
spatial_res2b_branch2a: Conv2D
input: (None, 55, 55, 256)
output: (None, 55, 55, 64)
spatial_bn2b_branch2a: BatchNormalization
input: (None, 55, 55, 64)
output: (None, 55, 55, 64)
spatial_activation_330: Activation
input: (None, 55, 55, 64)
output: (None, 55, 55, 64)
spatial_res2b_branch2b: Conv2D
input: (None, 55, 55, 64)
output: (None, 55, 55, 64)
spatial_bn2b_branch2b: BatchNormalization
input: (None, 55, 55, 64)
output: (None, 55, 55, 64)
spatial_activation_331: Activation
input: (None, 55, 55, 64)
output: (None, 55, 55, 64)
spatial_res2b_branch2c: Conv2D
input: (None, 55, 55, 64)
output: (None, 55, 55, 256)
spatial_bn2b_branch2c: BatchNormalization
input: (None, 55, 55, 256)
output: (None, 55, 55, 256)
spatial_add_82: Add
input: [(None, 55, 55, 256), (None, 55, 55, 256)]
output: (None, 55, 55, 256)
spatial_activation_332: Activation
input: (None, 55, 55, 256)
output: (None, 55, 55, 256)
spatial_res2c_branch2a: Conv2D
input: (None, 55, 55, 256)
output: (None, 55, 55, 64)
spatial_bn2c_branch2a: BatchNormalization
input: (None, 55, 55, 64)
output: (None, 55, 55, 64)
input: (None, 55, 55, 64)
pose_input_2: InputLayer
input: (None, 224, 224, 30)
output: (None, 224, 224, 30)
pose_zero_padding2d_2: ZeroPadding2D
input: (None, 224, 224, 30)
output: (None, 230, 230, 30)
pose_conv1: Conv2D
input: (None, 230, 230, 30)
output: (None, 112, 112, 64)
pose_bn_conv1: BatchNormalization
input: (None, 112, 112, 64)
output: (None, 112, 112, 64)
pose_activation_50: Activation
input: (None, 112, 112, 64)
output: (None, 112, 112, 64)
pose_max_pooling2d_2: MaxPooling2D
input: (None, 112, 112, 64)
output: (None, 55, 55, 64)
pose_res2a_branch2a: Conv2D
input: (None, 55, 55, 64)
output: (None, 55, 55, 64)
pose_res2a_branch1: Conv2D
input: (None, 55, 55, 64)
output: (None, 55, 55, 256)
pose_bn2a_branch2a: BatchNormalization
input: (None, 55, 55, 64)
output: (None, 55, 55, 64)
pose_activation_51: Activation
input: (None, 55, 55, 64)
output: (None, 55, 55, 64)
pose_bn2a_branch1: BatchNormalization
input: (None, 55, 55, 256)
output: (None, 55, 55, 256)
pose_res2a_branch2b: Conv2D
input: (None, 55, 55, 64)
output: (None, 55, 55, 64)
pose_bn2a_branch2b: BatchNormalization
input: (None, 55, 55, 64)
output: (None, 55, 55, 64)
pose_activation_52: Activation
input: (None, 55, 55, 64)
output: (None, 55, 55, 64)
pose_res2a_branch2c: Conv2D
input: (None, 55, 55, 64)
output: (None, 55, 55, 256)
pose_bn2a_branch2c: BatchNormalization
input: (None, 55, 55, 256)
output: (None, 55, 55, 256)
pose_add_17: Add
input: [(None, 55, 55, 256), (None, 55, 55, 256)]
output: (None, 55, 55, 256)
pose_activation_53: Activation
input: (None, 55, 55, 256)
output: (None, 55, 55, 256)
pose_res2b_branch2a: Conv2D
input: (None, 55, 55, 256)
output: (None, 55, 55, 64)
pose_bn2b_branch2a: BatchNormalization
input: (None, 55, 55, 64)
output: (None, 55, 55, 64)
pose_activation_54: Activation
input: (None, 55, 55, 64)
output: (None, 55, 55, 64)
pose_res2b_branch2b: Conv2D
input: (None, 55, 55, 64)
output: (None, 55, 55, 64)
pose_bn2b_branch2b: BatchNormalization
input: (None, 55, 55, 64)
output: (None, 55, 55, 64)
pose_activation_55: Activation
input: (None, 55, 55, 64)
output: (None, 55, 55, 64)
pose_res2b_branch2c: Conv2D
input: (None, 55, 55, 64)
output: (None, 55, 55, 256)
pose_bn2b_branch2c: BatchNormalization
input: (None, 55, 55, 256)
output: (None, 55, 55, 256)
pose_add_18: Add
input: [(None, 55, 55, 256), (None, 55, 55, 256)]
output: (None, 55, 55, 256)
pose_activation_56: Activation
input: (None, 55, 55, 256)
output: (None, 55, 55, 256)
pose_res2c_branch2a: Conv2D
input: (None, 55, 55, 256)
output: (None, 55, 55, 64)
pose_bn2c_branch2a: BatchNormalization
input: (None, 55, 55, 64)
output: (None, 55, 55, 64)
input: (None, 55, 55, 64)

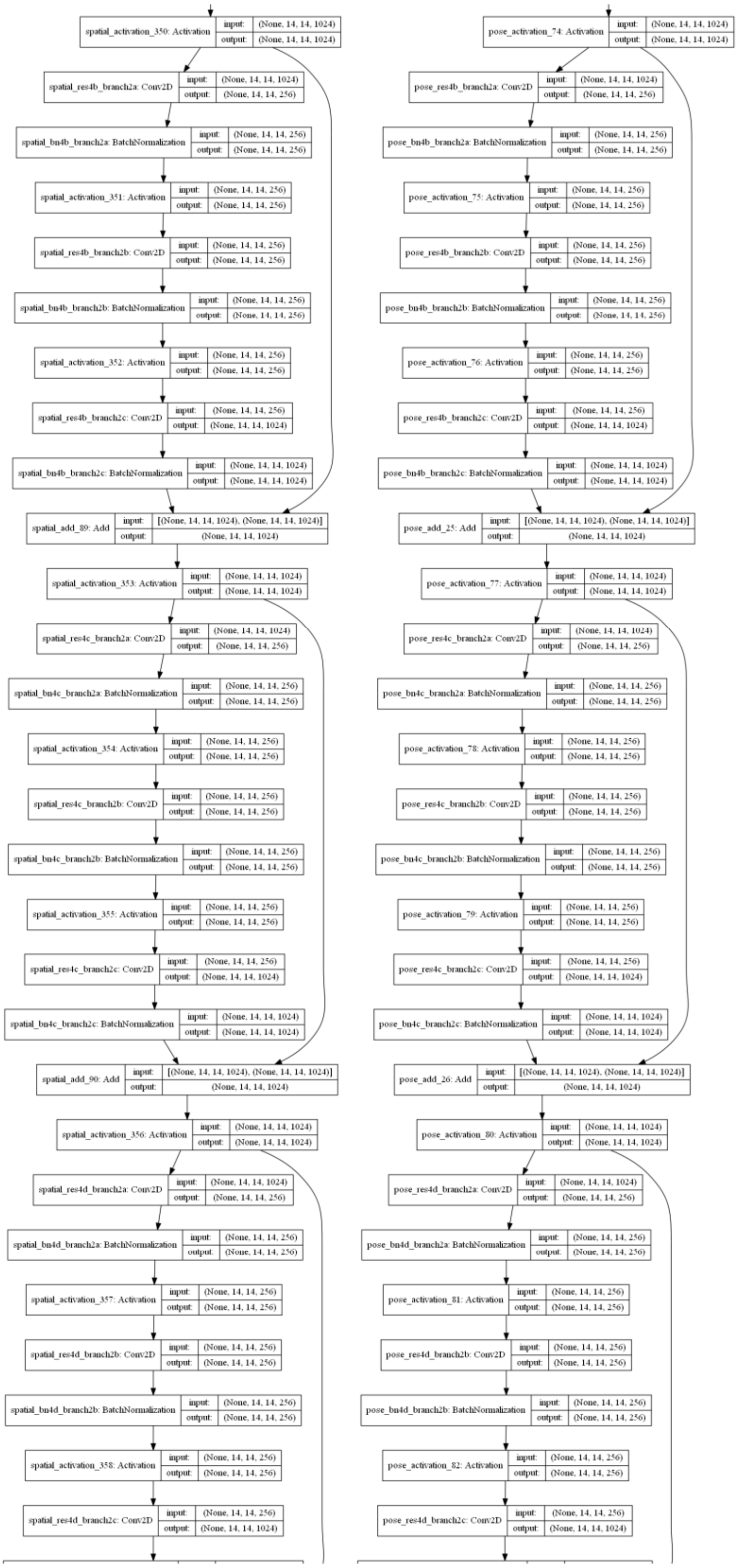
Appendix


spatial_activation_333: Activation
output: (None, 55, 55, 64)
spatial_res2c_branch2b: Conv2D
input: (None, 55, 55, 64)
output: (None, 55, 55, 64)
spatial_bn2c_branch2b: BatchNormalization
input: (None, 55, 55, 64)
output: (None, 55, 55, 64)
spatial_activation_334: Activation
input: (None, 55, 55, 64)
output: (None, 55, 55, 64)
spatial_res2c_branch2c: Conv2D
input: (None, 55, 55, 64)
output: (None, 55, 55, 256)
spatial_bn2c_branch2c: BatchNormalization
input: (None, 55, 55, 256)
output: (None, 55, 55, 256)
spatial_add_83: Add
input: [(None, 55, 55, 256), (None, 55, 55, 256)]
output: (None, 55, 55, 256)
spatial_activation_335: Activation
input: (None, 55, 55, 256)
output: (None, 55, 55, 256)
spatial_res3a_branch2a: Conv2D
input: (None, 55, 55, 256)
output: (None, 28, 28, 128)
spatial_bn3a_branch2a: BatchNormalization
input: (None, 28, 28, 128)
output: (None, 28, 28, 128)
spatial_res3a_branch1: Conv2D
input: (None, 55, 55, 256)
output: (None, 28, 28, 512)
spatial_activation_336: Activation
input: (None, 28, 28, 128)
output: (None, 28, 28, 128)
spatial_res3a_branch2b: Conv2D
input: (None, 28, 28, 128)
output: (None, 28, 28, 128)
spatial_bn3a_branch1: BatchNormalization
input: (None, 28, 28, 512)
output: (None, 28, 28, 512)
spatial_bn3a_branch2b: BatchNormalization
input: (None, 28, 28, 128)
output: (None, 28, 28, 128)
spatial_activation_337: Activation
input: (None, 28, 28, 128)
output: (None, 28, 28, 128)
spatial_res3a_branch2c: Conv2D
input: (None, 28, 28, 128)
output: (None, 28, 28, 512)
spatial_bn3a_branch2c: BatchNormalization
input: (None, 28, 28, 512)
output: (None, 28, 28, 512)
spatial_add_84: Add
input: [(None, 28, 28, 512), (None, 28, 28, 512)]
output: (None, 28, 28, 512)
spatial_activation_338: Activation
input: (None, 28, 28, 512)
output: (None, 28, 28, 512)
spatial_res3b_branch2a: Conv2D
input: (None, 28, 28, 512)
output: (None, 28, 28, 128)
spatial_bn3b_branch2a: BatchNormalization
input: (None, 28, 28, 128)
output: (None, 28, 28, 128)
spatial_activation_339: Activation
input: (None, 28, 28, 128)
output: (None, 28, 28, 128)
spatial_res3b_branch2b: Conv2D
input: (None, 28, 28, 128)
output: (None, 28, 28, 128)
spatial_bn3b_branch2b: BatchNormalization
input: (None, 28, 28, 128)
output: (None, 28, 28, 128)
spatial_activation_340: Activation
input: (None, 28, 28, 128)
output: (None, 28, 28, 128)
spatial_res3b_branch2c: Conv2D
input: (None, 28, 28, 128)
output: (None, 28, 28, 512)
spatial_bn3b_branch2c: BatchNormalization
input: (None, 28, 28, 512)
output: (None, 28, 28, 512)
spatial_add_85: Add
input: [(None, 28, 28, 512), (None, 28, 28, 512)]
output: (None, 28, 28, 512)
spatial_activation_341: Activation
input: (None, 28, 28, 512)
output: (None, 28, 28, 512)
spatial_res3c_branch2a: Conv2D
input: (None, 28, 28, 512)
output: (None, 28, 28, 128)
pose_activation_57: Activation
output: (None, 55, 55, 64)
pose_res2c_branch2b: Conv2D
input: (None, 55, 55, 64)
output: (None, 55, 55, 64)
pose_bn2c_branch2b: BatchNormalization
input: (None, 55, 55, 64)
output: (None, 55, 55, 64)
pose_activation_58: Activation
input: (None, 55, 55, 64)
output: (None, 55, 55, 64)
pose_res2c_branch2c: Conv2D
input: (None, 55, 55, 64)
output: (None, 55, 55, 256)
pose_bn2c_branch2c: BatchNormalization
input: (None, 55, 55, 256)
output: (None, 55, 55, 256)
pose_add_19: Add
input: [(None, 55, 55, 256), (None, 55, 55, 256)]
output: (None, 55, 55, 256)
pose_activation_59: Activation
input: (None, 55, 55, 256)
output: (None, 55, 55, 256)
pose_res3a_branch2a: Conv2D
input: (None, 55, 55, 256)
output: (None, 28, 28, 128)
pose_res3a_branch1: Conv2D
input: (None, 55, 55, 256)
output: (None, 28, 28, 512)
pose_bn3a_branch2a: BatchNormalization
input: (None, 28, 28, 128)
output: (None, 28, 28, 128)
pose_activation_60: Activation
input: (None, 28, 28, 128)
output: (None, 28, 28, 128)
pose_bn3a_branch1: BatchNormalization
input: (None, 28, 28, 512)
output: (None, 28, 28, 512)
pose_res3a_branch2b: Conv2D
input: (None, 28, 28, 128)
output: (None, 28, 28, 128)
pose_bn3a_branch2b: BatchNormalization
input: (None, 28, 28, 128)
output: (None, 28, 28, 128)
pose_activation_61: Activation
input: (None, 28, 28, 128)
output: (None, 28, 28, 128)
pose_res3a_branch2c: Conv2D
input: (None, 28, 28, 128)
output: (None, 28, 28, 512)
pose_bn3a_branch2c: BatchNormalization
input: (None, 28, 28, 512)
output: (None, 28, 28, 512)
pose_add_20: Add
input: [(None, 28, 28, 512), (None, 28, 28, 512)]
output: (None, 28, 28, 512)
pose_activation_62: Activation
input: (None, 28, 28, 512)
output: (None, 28, 28, 512)
pose_res3b_branch2a: Conv2D
input: (None, 28, 28, 512)
output: (None, 28, 28, 128)
pose_bn3b_branch2a: BatchNormalization
input: (None, 28, 28, 128)
output: (None, 28, 28, 128)
pose_activation_63: Activation
input: (None, 28, 28, 128)
output: (None, 28, 28, 128)
pose_res3b_branch2b: Conv2D
input: (None, 28, 28, 128)
output: (None, 28, 28, 128)
pose_bn3b_branch2b: BatchNormalization
input: (None, 28, 28, 128)
output: (None, 28, 28, 128)
pose_activation_64: Activation
input: (None, 28, 28, 128)
output: (None, 28, 28, 128)
pose_res3b_branch2c: Conv2D
input: (None, 28, 28, 128)
output: (None, 28, 28, 512)
pose_bn3b_branch2c: BatchNormalization
input: (None, 28, 28, 512)
output: (None, 28, 28, 512)
pose_add_21: Add
input: [(None, 28, 28, 512), (None, 28, 28, 512)]
output: (None, 28, 28, 512)
pose_activation_65: Activation
input: (None, 28, 28, 512)
output: (None, 28, 28, 512)
pose_res3c_branch2a: Conv2D
input: (None, 28, 28, 512)
output: (None, 28, 28, 128)

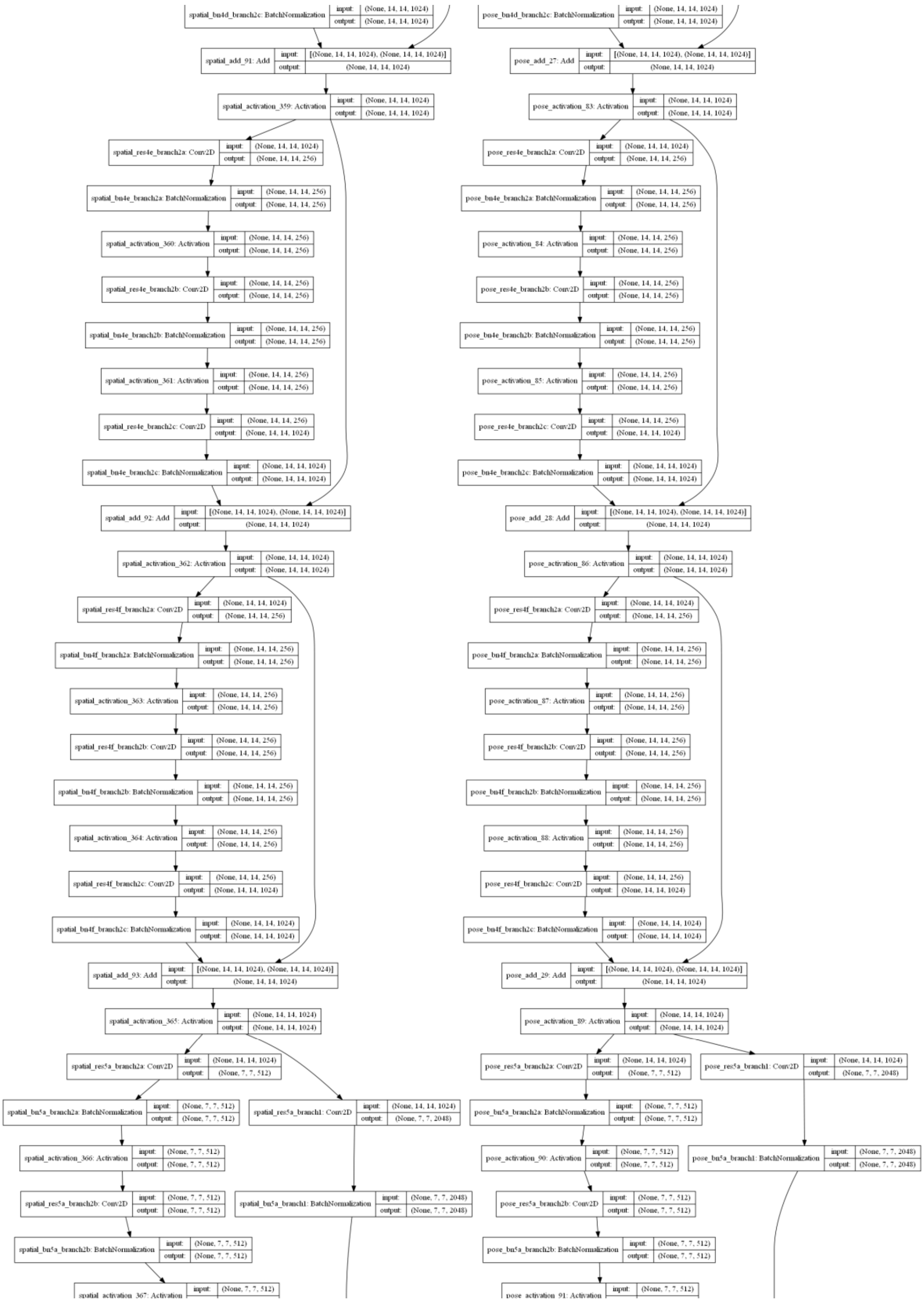
Appendix


spatial_bn3c_branch2a: BatchNormalization
input: (None, 28, 28, 128)
output: (None, 28, 28, 128)
spatial_activation_342: Activation
input: (None, 28, 28, 128)
output: (None, 28, 28, 128)
spatial_res3c_branch2b: Conv2D
input: (None, 28, 28, 128)
output: (None, 28, 28, 128)
spatial_bn3c_branch2b: BatchNormalization
input: (None, 28, 28, 128)
output: (None, 28, 28, 128)
spatial_activation_343: Activation
input: (None, 28, 28, 128)
output: (None, 28, 28, 128)
spatial_res3c_branch2c: Conv2D
input: (None, 28, 28, 128)
output: (None, 28, 28, 512)
spatial_bn3c_branch2c: BatchNormalization
input: (None, 28, 28, 512)
output: (None, 28, 28, 512)
spatial_add_86: Add
input: [(None, 28, 28, 512), (None, 28, 28, 512)]
output: (None, 28, 28, 512)
spatial_activation_344: Activation
input: (None, 28, 28, 512)
output: (None, 28, 28, 512)
spatial_res3d_branch2a: Conv2D
input: (None, 28, 28, 512)
output: (None, 28, 28, 128)
spatial_bn3d_branch2a: BatchNormalization
input: (None, 28, 28, 128)
output: (None, 28, 28, 128)
spatial_activation_345: Activation
input: (None, 28, 28, 128)
output: (None, 28, 28, 128)
spatial_res3d_branch2b: Conv2D
input: (None, 28, 28, 128)
output: (None, 28, 28, 128)
spatial_bn3d_branch2b: BatchNormalization
input: (None, 28, 28, 128)
output: (None, 28, 28, 128)
spatial_activation_346: Activation
input: (None, 28, 28, 128)
output: (None, 28, 28, 128)
spatial_res3d_branch2c: Conv2D
input: (None, 28, 28, 128)
output: (None, 28, 28, 512)
spatial_bn3d_branch2c: BatchNormalization
input: (None, 28, 28, 512)
output: (None, 28, 28, 512)
spatial_add_87: Add
input: [(None, 28, 28, 512), (None, 28, 28, 512)]
output: (None, 28, 28, 512)
spatial_activation_347: Activation
input: (None, 28, 28, 512)
output: (None, 28, 28, 512)
spatial_res4a_branch2a: Conv2D
input: (None, 28, 28, 512)
output: (None, 14, 14, 256)
spatial_bn4a_branch2a: BatchNormalization
input: (None, 14, 14, 256)
output: (None, 14, 14, 256)
spatial_res4a_branch1: Conv2D
input: (None, 28, 28, 512)
output: (None, 14, 14, 1024)
spatial_activation_348: Activation
input: (None, 14, 14, 256)
output: (None, 14, 14, 256)
spatial_res4a_branch2b: Conv2D
input: (None, 14, 14, 256)
output: (None, 14, 14, 256)
spatial_bn4a_branch1: BatchNormalization
input: (None, 14, 14, 1024)
output: (None, 14, 14, 1024)
spatial_bn4a_branch2b: BatchNormalization
input: (None, 14, 14, 256)
output: (None, 14, 14, 256)
spatial_activation_349: Activation
input: (None, 14, 14, 256)
output: (None, 14, 14, 256)
spatial_res4a_branch2c: Conv2D
input: (None, 14, 14, 256)
output: (None, 14, 14, 1024)
spatial_bn4a_branch2c: BatchNormalization
input: (None, 14, 14, 1024)
output: (None, 14, 14, 1024)
spatial_add_88: Add
input: [(None, 14, 14, 1024), (None, 14, 14, 1024)]
output: (None, 14, 14, 1024)
pose_bn3c_branch2a: BatchNormalization
input: (None, 28, 28, 128)
output: (None, 28, 28, 128)
pose_activation_66: Activation
input: (None, 28, 28, 128)
output: (None, 28, 28, 128)
pose_res3c_branch2b: Conv2D
input: (None, 28, 28, 128)
output: (None, 28, 28, 128)
pose_bn3c_branch2b: BatchNormalization
input: (None, 28, 28, 128)
output: (None, 28, 28, 128)
pose_activation_67: Activation
input: (None, 28, 28, 128)
output: (None, 28, 28, 128)
pose_res3c_branch2c: Conv2D
input: (None, 28, 28, 128)
output: (None, 28, 28, 512)
pose_bn3c_branch2c: BatchNormalization
input: (None, 28, 28, 512)
output: (None, 28, 28, 512)
pose_add_22: Add
input: [(None, 28, 28, 512), (None, 28, 28, 512)]
output: (None, 28, 28, 512)
pose_activation_68: Activation
input: (None, 28, 28, 512)
output: (None, 28, 28, 512)
pose_res3d_branch2a: Conv2D
input: (None, 28, 28, 512)
output: (None, 28, 28, 128)
pose_bn3d_branch2a: BatchNormalization
input: (None, 28, 28, 128)
output: (None, 28, 28, 128)
pose_activation_69: Activation
input: (None, 28, 28, 128)
output: (None, 28, 28, 128)
pose_res3d_branch2b: Conv2D
input: (None, 28, 28, 128)
output: (None, 28, 28, 128)
pose_bn3d_branch2b: BatchNormalization
input: (None, 28, 28, 128)
output: (None, 28, 28, 128)
pose_activation_70: Activation
input: (None, 28, 28, 128)
output: (None, 28, 28, 128)
pose_res3d_branch2c: Conv2D
input: (None, 28, 28, 128)
output: (None, 28, 28, 512)
pose_bn3d_branch2c: BatchNormalization
input: (None, 28, 28, 512)
output: (None, 28, 28, 512)
pose_add_23: Add
input: [(None, 28, 28, 512), (None, 28, 28, 512)]
output: (None, 28, 28, 512)
pose_activation_71: Activation
input: (None, 28, 28, 512)
output: (None, 28, 28, 512)
pose_res4a_branch2a: Conv2D
input: (None, 28, 28, 512)
output: (None, 14, 14, 256)
pose_res4a_branch1: Conv2D
input: (None, 28, 28, 512)
output: (None, 14, 14, 1024)
pose_bn4a_branch2a: BatchNormalization
input: (None, 14, 14, 256)
output: (None, 14, 14, 256)
pose_activation_72: Activation
input: (None, 14, 14, 256)
output: (None, 14, 14, 256)
pose_bn4a_branch1: BatchNormalization
input: (None, 14, 14, 1024)
output: (None, 14, 14, 1024)
pose_res4a_branch2b: Conv2D
input: (None, 14, 14, 256)
output: (None, 14, 14, 256)
pose_bn4a_branch2b: BatchNormalization
input: (None, 14, 14, 256)
output: (None, 14, 14, 256)
pose_activation_73: Activation
input: (None, 14, 14, 256)
output: (None, 14, 14, 256)
pose_res4a_branch2c: Conv2D
input: (None, 14, 14, 256)
output: (None, 14, 14, 1024)
pose_bn4a_branch2c: BatchNormalization
input: (None, 14, 14, 1024)
output: (None, 14, 14, 1024)
pose_add_24: Add
input: [(None, 14, 14, 1024), (None, 14, 14, 1024)]
output: (None, 14, 14, 1024)

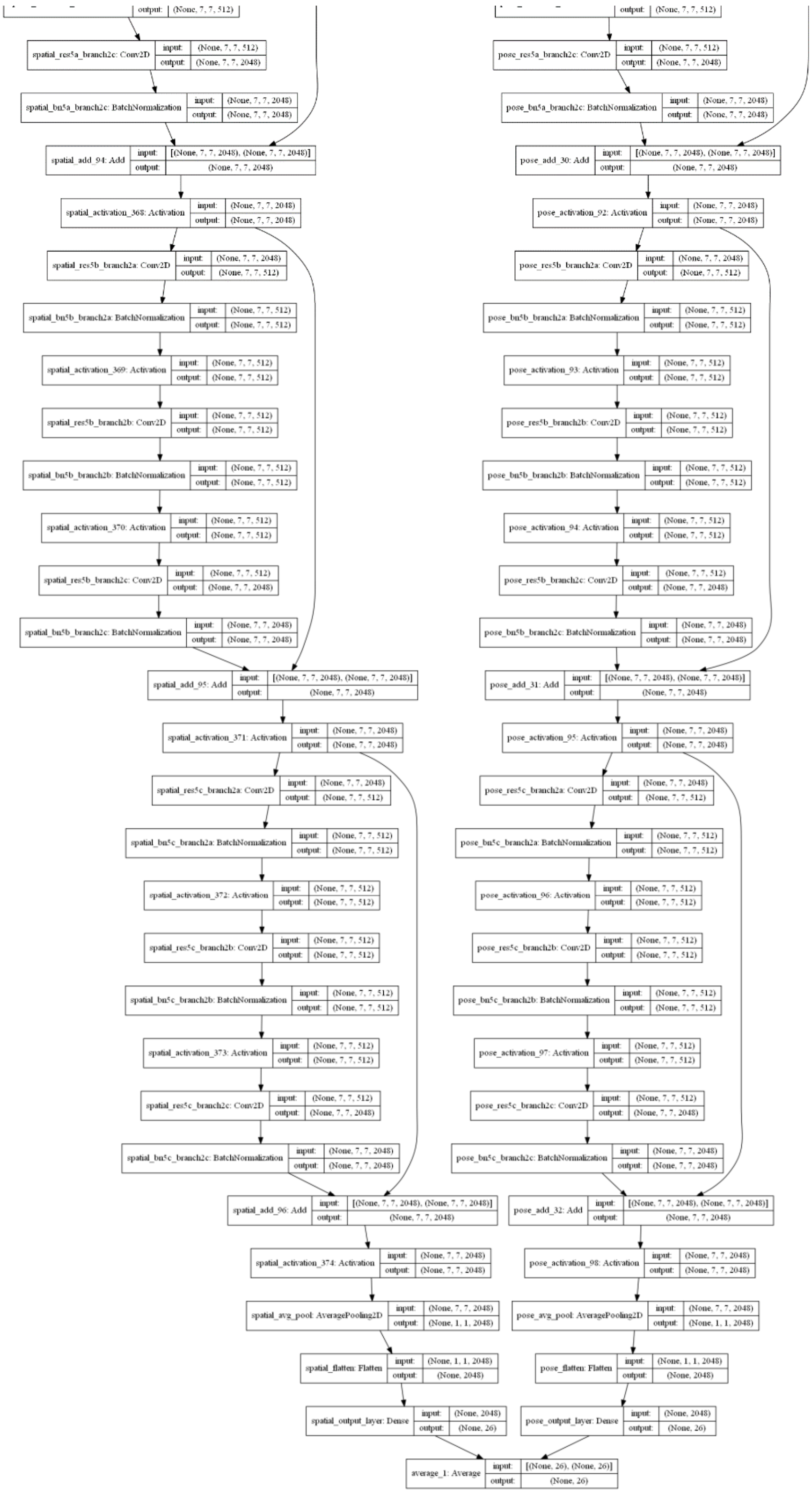
Appendix


spatial_activation_350: Activation input: (None, 14, 14, 1024) output: (None, 14, 14, 1024)
spatial_res4b_branch2a: Conv2D input: (None, 14, 14, 1024) output: (None, 14, 14, 256)
spatial_bn4b_branch2a: BatchNormalization input: (None, 14, 14, 256) output: (None, 14, 14, 256)
spatial_activation_351: Activation input: (None, 14, 14, 256) output: (None, 14, 14, 256)
spatial_res4b_branch2b: Conv2D input: (None, 14, 14, 256) output: (None, 14, 14, 256)
spatial_bn4b_branch2b: BatchNormalization input: (None, 14, 14, 256) output: (None, 14, 14, 256)
spatial_activation_352: Activation input: (None, 14, 14, 256) output: (None, 14, 14, 256)
spatial_res4b_branch2c: Conv2D input: (None, 14, 14, 256) output: (None, 14, 14, 1024)
spatial_bn4b_branch2c: BatchNormalization input: (None, 14, 14, 1024) output: (None, 14, 14, 1024)
spatial_add_89: Add input: [(None, 14, 14, 1024), (None, 14, 14, 1024)] output: (None, 14, 14, 1024)
spatial_activation_353: Activation input: (None, 14, 14, 1024) output: (None, 14, 14, 1024)
spatial_res4c_branch2a: Conv2D input: (None, 14, 14, 1024) output: (None, 14, 14, 256)
spatial_bn4c_branch2a: BatchNormalization input: (None, 14, 14, 256) output: (None, 14, 14, 256)
spatial_activation_354: Activation input: (None, 14, 14, 256) output: (None, 14, 14, 256)
spatial_res4c_branch2b: Conv2D input: (None, 14, 14, 256) output: (None, 14, 14, 256)
spatial_bn4c_branch2b: BatchNormalization input: (None, 14, 14, 256) output: (None, 14, 14, 256)
spatial_activation_355: Activation input: (None, 14, 14, 256) output: (None, 14, 14, 256)
spatial_res4c_branch2c: Conv2D input: (None, 14, 14, 256) output: (None, 14, 14, 1024)
spatial_bn4c_branch2c: BatchNormalization input: (None, 14, 14, 1024) output: (None, 14, 14, 1024)
spatial_add_90: Add input: [(None, 14, 14, 1024), (None, 14, 14, 1024)] output: (None, 14, 14, 1024)
spatial_activation_356: Activation input: (None, 14, 14, 1024) output: (None, 14, 14, 1024)
spatial_res4d_branch2a: Conv2D input: (None, 14, 14, 1024) output: (None, 14, 14, 256)
spatial_bn4d_branch2a: BatchNormalization input: (None, 14, 14, 256) output: (None, 14, 14, 256)
spatial_activation_357: Activation input: (None, 14, 14, 256) output: (None, 14, 14, 256)
spatial_res4d_branch2b: Conv2D input: (None, 14, 14, 256) output: (None, 14, 14, 256)
spatial_bn4d_branch2b: BatchNormalization input: (None, 14, 14, 256) output: (None, 14, 14, 256)
spatial_activation_358: Activation input: (None, 14, 14, 256) output: (None, 14, 14, 256)
spatial_res4d_branch2c: Conv2D input: (None, 14, 14, 256) output: (None, 14, 14, 1024)
pose_activation_74: Activation input: (None, 14, 14, 1024) output: (None, 14, 14, 1024)
pose_res4b_branch2a: Conv2D input: (None, 14, 14, 1024) output: (None, 14, 14, 256)
pose_bn4b_branch2a: BatchNormalization input: (None, 14, 14, 256) output: (None, 14, 14, 256)
pose_activation_75: Activation input: (None, 14, 14, 256) output: (None, 14, 14, 256)
pose_res4b_branch2b: Conv2D input: (None, 14, 14, 256) output: (None, 14, 14, 256)
pose_bn4b_branch2b: BatchNormalization input: (None, 14, 14, 256) output: (None, 14, 14, 256)
pose_activation_76: Activation input: (None, 14, 14, 256) output: (None, 14, 14, 256)
pose_res4b_branch2c: Conv2D input: (None, 14, 14, 256) output: (None, 14, 14, 1024)
pose_bn4b_branch2c: BatchNormalization input: (None, 14, 14, 1024) output: (None, 14, 14, 1024)
pose_add_25: Add input: [(None, 14, 14, 1024), (None, 14, 14, 1024)] output: (None, 14, 14, 1024)
pose_activation_77: Activation input: (None, 14, 14, 1024) output: (None, 14, 14, 1024)
pose_res4c_branch2a: Conv2D input: (None, 14, 14, 1024) output: (None, 14, 14, 256)
pose_bn4c_branch2a: BatchNormalization input: (None, 14, 14, 256) output: (None, 14, 14, 256)
pose_activation_78: Activation input: (None, 14, 14, 256) output: (None, 14, 14, 256)
pose_res4c_branch2b: Conv2D input: (None, 14, 14, 256) output: (None, 14, 14, 256)
pose_bn4c_branch2b: BatchNormalization input: (None, 14, 14, 256) output: (None, 14, 14, 256)
pose_activation_79: Activation input: (None, 14, 14, 256) output: (None, 14, 14, 256)
pose_res4c_branch2c: Conv2D input: (None, 14, 14, 256) output: (None, 14, 14, 1024)
pose_bn4c_branch2c: BatchNormalization input: (None, 14, 14, 1024) output: (None, 14, 14, 1024)
pose_add_26: Add input: [(None, 14, 14, 1024), (None, 14, 14, 1024)] output: (None, 14, 14, 1024)
pose_activation_80: Activation input: (None, 14, 14, 1024) output: (None, 14, 14, 1024)
pose_res4d_branch2a: Conv2D input: (None, 14, 14, 1024) output: (None, 14, 14, 256)
pose_bn4d_branch2a: BatchNormalization input: (None, 14, 14, 256) output: (None, 14, 14, 256)
pose_activation_81: Activation input: (None, 14, 14, 256) output: (None, 14, 14, 256)
pose_res4d_branch2b: Conv2D input: (None, 14, 14, 256) output: (None, 14, 14, 256)
pose_bn4d_branch2b: BatchNormalization input: (None, 14, 14, 256) output: (None, 14, 14, 256)
pose_activation_82: Activation input: (None, 14, 14, 256) output: (None, 14, 14, 256)
pose_res4d_branch2c: Conv2D input: (None, 14, 14, 256) output: (None, 14, 14, 1024)

Appendix

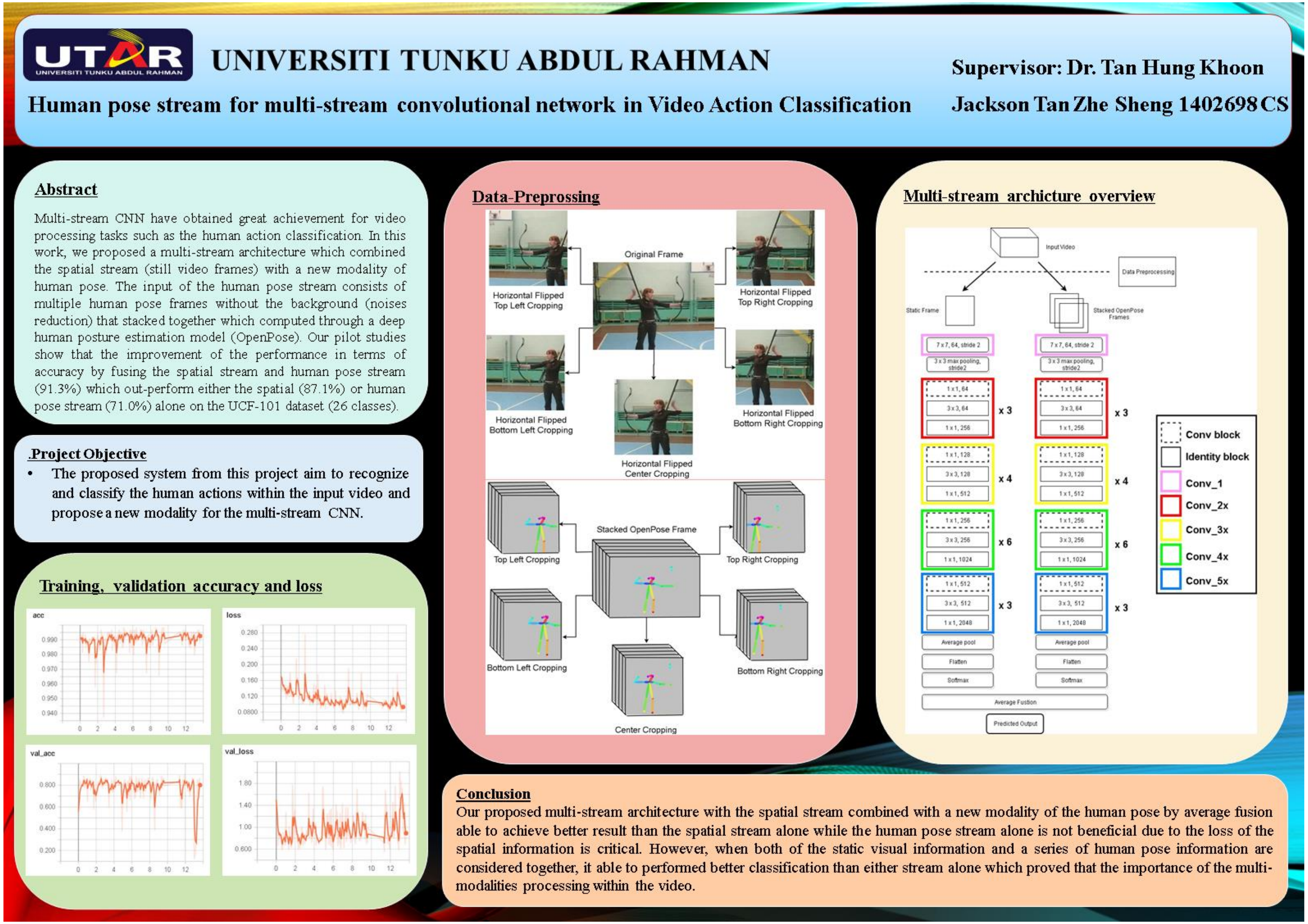

spatial_bn4d_branch2c: BatchNormalization input: (None, 14, 14, 1024) output: (None, 14, 14, 1024)
spatial_add_91: Add input: [(None, 14, 14, 1024), (None, 14, 14, 1024)] output: (None, 14, 14, 1024)
spatial_activation_359: Activation input: (None, 14, 14, 1024) output: (None, 14, 14, 1024)
spatial_res4e_branch2a: Conv2D input: (None, 14, 14, 1024) output: (None, 14, 14, 256)
spatial_bn4e_branch2a: BatchNormalization input: (None, 14, 14, 256) output: (None, 14, 14, 256)
spatial_activation_360: Activation input: (None, 14, 14, 256) output: (None, 14, 14, 256)
spatial_res4e_branch2b: Conv2D input: (None, 14, 14, 256) output: (None, 14, 14, 256)
spatial_bn4e_branch2b: BatchNormalization input: (None, 14, 14, 256) output: (None, 14, 14, 256)
spatial_activation_361: Activation input: (None, 14, 14, 256) output: (None, 14, 14, 256)
spatial_res4e_branch2c: Conv2D input: (None, 14, 14, 256) output: (None, 14, 14, 1024)
spatial_bn4e_branch2c: BatchNormalization input: (None, 14, 14, 1024) output: (None, 14, 14, 1024)
spatial_add_92: Add input: [(None, 14, 14, 1024), (None, 14, 14, 1024)] output: (None, 14, 14, 1024)
spatial_activation_362: Activation input: (None, 14, 14, 1024) output: (None, 14, 14, 1024)
spatial_res4f_branch2a: Conv2D input: (None, 14, 14, 1024) output: (None, 14, 14, 256)
spatial_bn4f_branch2a: BatchNormalization input: (None, 14, 14, 256) output: (None, 14, 14, 256)
spatial_activation_363: Activation input: (None, 14, 14, 256) output: (None, 14, 14, 256)
spatial_res4f_branch2b: Conv2D input: (None, 14, 14, 256) output: (None, 14, 14, 256)
spatial_bn4f_branch2b: BatchNormalization input: (None, 14, 14, 256) output: (None, 14, 14, 256)
spatial_activation_364: Activation input: (None, 14, 14, 256) output: (None, 14, 14, 256)
spatial_res4f_branch2c: Conv2D input: (None, 14, 14, 256) output: (None, 14, 14, 1024)
spatial_bn4f_branch2c: BatchNormalization input: (None, 14, 14, 1024) output: (None, 14, 14, 1024)
spatial_add_93: Add input: [(None, 14, 14, 1024), (None, 14, 14, 1024)] output: (None, 14, 14, 1024)
spatial_activation_365: Activation input: (None, 14, 14, 1024) output: (None, 14, 14, 1024)
spatial_res5a_branch2a: Conv2D input: (None, 14, 14, 1024) output: (None, 7, 7, 512)
spatial_bn5a_branch2a: BatchNormalization input: (None, 7, 7, 512) output: (None, 7, 7, 512)
spatial_res5a_branch1: Conv2D input: (None, 14, 14, 1024) output: (None, 7, 7, 2048)
spatial_activation_366: Activation input: (None, 7, 7, 512) output: (None, 7, 7, 512)
spatial_res5a_branch2b: Conv2D input: (None, 7, 7, 512) output: (None, 7, 7, 512)
spatial_bn5a_branch1: BatchNormalization input: (None, 7, 7, 2048) output: (None, 7, 7, 2048)
spatial_bn5a_branch2b: BatchNormalization input: (None, 7, 7, 512) output: (None, 7, 7, 512)
spatial_activation_367: Activation input: (None, 7, 7, 512)
pose_bn4d_branch2c: BatchNormalization input: (None, 14, 14, 1024) output: (None, 14, 14, 1024)
pose_add_27: Add input: [(None, 14, 14, 1024), (None, 14, 14, 1024)] output: (None, 14, 14, 1024)
pose_activation_83: Activation input: (None, 14, 14, 1024) output: (None, 14, 14, 1024)
pose_res4e_branch2a: Conv2D input: (None, 14, 14, 1024) output: (None, 14, 14, 256)
pose_bn4e_branch2a: BatchNormalization input: (None, 14, 14, 256) output: (None, 14, 14, 256)
pose_activation_84: Activation input: (None, 14, 14, 256) output: (None, 14, 14, 256)
pose_res4e_branch2b: Conv2D input: (None, 14, 14, 256) output: (None, 14, 14, 256)
pose_bn4e_branch2b: BatchNormalization input: (None, 14, 14, 256) output: (None, 14, 14, 256)
pose_activation_85: Activation input: (None, 14, 14, 256) output: (None, 14, 14, 256)
pose_res4e_branch2c: Conv2D input: (None, 14, 14, 256) output: (None, 14, 14, 1024)
pose_bn4e_branch2c: BatchNormalization input: (None, 14, 14, 1024) output: (None, 14, 14, 1024)
pose_add_28: Add input: [(None, 14, 14, 1024), (None, 14, 14, 1024)] output: (None, 14, 14, 1024)
pose_activation_86: Activation input: (None, 14, 14, 1024) output: (None, 14, 14, 1024)
pose_res4f_branch2a: Conv2D input: (None, 14, 14, 1024) output: (None, 14, 14, 256)
pose_bn4f_branch2a: BatchNormalization input: (None, 14, 14, 256) output: (None, 14, 14, 256)
pose_activation_87: Activation input: (None, 14, 14, 256) output: (None, 14, 14, 256)
pose_res4f_branch2b: Conv2D input: (None, 14, 14, 256) output: (None, 14, 14, 256)
pose_bn4f_branch2b: BatchNormalization input: (None, 14, 14, 256) output: (None, 14, 14, 256)
pose_activation_88: Activation input: (None, 14, 14, 256) output: (None, 14, 14, 256)
pose_res4f_branch2c: Conv2D input: (None, 14, 14, 256) output: (None, 14, 14, 1024)
pose_bn4f_branch2c: BatchNormalization input: (None, 14, 14, 1024) output: (None, 14, 14, 1024)
pose_add_29: Add input: [(None, 14, 14, 1024), (None, 14, 14, 1024)] output: (None, 14, 14, 1024)
pose_activation_89: Activation input: (None, 14, 14, 1024) output: (None, 14, 14, 1024)
pose_res5a_branch2a: Conv2D input: (None, 14, 14, 1024) output: (None, 7, 7, 512)
pose_res5a_branch1: Conv2D input: (None, 14, 14, 1024) output: (None, 7, 7, 2048)
pose_bn5a_branch2a: BatchNormalization input: (None, 7, 7, 512) output: (None, 7, 7, 512)
pose_activation_90: Activation input: (None, 7, 7, 512) output: (None, 7, 7, 512)
pose_bn5a_branch1: BatchNormalization input: (None, 7, 7, 2048) output: (None, 7, 7, 2048)
pose_res5a_branch2b: Conv2D input: (None, 7, 7, 512) output: (None, 7, 7, 512)
pose_bn5a_branch2b: BatchNormalization input: (None, 7, 7, 512) output: (None, 7, 7, 512)
pose_activation_91: Activation input: (None, 7, 7, 512)

Appendix


output: (None, 7, 7, 512)
spatial_res5a_branch2c: Conv2D
input: (None, 7, 7, 512)
output: (None, 7, 7, 2048)
spatial_bn5a_branch2c: BatchNormalization
input: (None, 7, 7, 2048)
output: (None, 7, 7, 2048)
spatial_add_94: Add
input: [(None, 7, 7, 2048), (None, 7, 7, 2048)]
output: (None, 7, 7, 2048)
spatial_activation_368: Activation
input: (None, 7, 7, 2048)
output: (None, 7, 7, 2048)
spatial_res5b_branch2a: Conv2D
input: (None, 7, 7, 2048)
output: (None, 7, 7, 512)
spatial_bn5b_branch2a: BatchNormalization
input: (None, 7, 7, 512)
output: (None, 7, 7, 512)
spatial_activation_369: Activation
input: (None, 7, 7, 512)
output: (None, 7, 7, 512)
spatial_res5b_branch2b: Conv2D
input: (None, 7, 7, 512)
output: (None, 7, 7, 512)
spatial_bn5b_branch2b: BatchNormalization
input: (None, 7, 7, 512)
output: (None, 7, 7, 512)
spatial_activation_370: Activation
input: (None, 7, 7, 512)
output: (None, 7, 7, 512)
spatial_res5b_branch2c: Conv2D
input: (None, 7, 7, 512)
output: (None, 7, 7, 2048)
spatial_bn5b_branch2c: BatchNormalization
input: (None, 7, 7, 2048)
output: (None, 7, 7, 2048)
spatial_add_95: Add
input: [(None, 7, 7, 2048), (None, 7, 7, 2048)]
output: (None, 7, 7, 2048)
spatial_activation_371: Activation
input: (None, 7, 7, 2048)
output: (None, 7, 7, 2048)
spatial_res5c_branch2a: Conv2D
input: (None, 7, 7, 2048)
output: (None, 7, 7, 512)
spatial_bn5c_branch2a: BatchNormalization
input: (None, 7, 7, 512)
output: (None, 7, 7, 512)
spatial_activation_372: Activation
input: (None, 7, 7, 512)
output: (None, 7, 7, 512)
spatial_res5c_branch2b: Conv2D
input: (None, 7, 7, 512)
output: (None, 7, 7, 512)
spatial_bn5c_branch2b: BatchNormalization
input: (None, 7, 7, 512)
output: (None, 7, 7, 512)
spatial_activation_373: Activation
input: (None, 7, 7, 512)
output: (None, 7, 7, 512)
spatial_res5c_branch2c: Conv2D
input: (None, 7, 7, 512)
output: (None, 7, 7, 2048)
spatial_bn5c_branch2c: BatchNormalization
input: (None, 7, 7, 2048)
output: (None, 7, 7, 2048)
spatial_add_96: Add
input: [(None, 7, 7, 2048), (None, 7, 7, 2048)]
output: (None, 7, 7, 2048)
spatial_activation_374: Activation
input: (None, 7, 7, 2048)
output: (None, 7, 7, 2048)
spatial_avg_pool: AveragePooling2D
input: (None, 7, 7, 2048)
output: (None, 1, 1, 2048)
spatial_flatten: Flatten
input: (None, 1, 1, 2048)
output: (None, 2048)
spatial_output_layer: Dense
input: (None, 2048)
output: (None, 26)
output: (None, 7, 7, 512)
pose_res5a_branch2c: Conv2D
input: (None, 7, 7, 512)
output: (None, 7, 7, 2048)
pose_bn5a_branch2c: BatchNormalization
input: (None, 7, 7, 2048)
output: (None, 7, 7, 2048)
pose_add_30: Add
input: [(None, 7, 7, 2048), (None, 7, 7, 2048)]
output: (None, 7, 7, 2048)
pose_activation_92: Activation
input: (None, 7, 7, 2048)
output: (None, 7, 7, 2048)
pose_res5b_branch2a: Conv2D
input: (None, 7, 7, 2048)
output: (None, 7, 7, 512)
pose_bn5b_branch2a: BatchNormalization
input: (None, 7, 7, 512)
output: (None, 7, 7, 512)
pose_activation_93: Activation
input: (None, 7, 7, 512)
output: (None, 7, 7, 512)
pose_res5b_branch2b: Conv2D
input: (None, 7, 7, 512)
output: (None, 7, 7, 512)
pose_bn5b_branch2b: BatchNormalization
input: (None, 7, 7, 512)
output: (None, 7, 7, 512)
pose_activation_94: Activation
input: (None, 7, 7, 512)
output: (None, 7, 7, 512)
pose_res5b_branch2c: Conv2D
input: (None, 7, 7, 512)
output: (None, 7, 7, 2048)
pose_bn5b_branch2c: BatchNormalization
input: (None, 7, 7, 2048)
output: (None, 7, 7, 2048)
pose_add_31: Add
input: [(None, 7, 7, 2048), (None, 7, 7, 2048)]
output: (None, 7, 7, 2048)
pose_activation_95: Activation
input: (None, 7, 7, 2048)
output: (None, 7, 7, 2048)
pose_res5c_branch2a: Conv2D
input: (None, 7, 7, 2048)
output: (None, 7, 7, 512)
pose_bn5c_branch2a: BatchNormalization
input: (None, 7, 7, 512)
output: (None, 7, 7, 512)
pose_activation_96: Activation
input: (None, 7, 7, 512)
output: (None, 7, 7, 512)
pose_res5c_branch2b: Conv2D
input: (None, 7, 7, 512)
output: (None, 7, 7, 512)
pose_bn5c_branch2b: BatchNormalization
input: (None, 7, 7, 512)
output: (None, 7, 7, 512)
pose_activation_97: Activation
input: (None, 7, 7, 512)
output: (None, 7, 7, 512)
pose_res5c_branch2c: Conv2D
input: (None, 7, 7, 512)
output: (None, 7, 7, 2048)
pose_bn5c_branch2c: BatchNormalization
input: (None, 7, 7, 2048)
output: (None, 7, 7, 2048)
pose_add_32: Add
input: [(None, 7, 7, 2048), (None, 7, 7, 2048)]
output: (None, 7, 7, 2048)
pose_activation_98: Activation
input: (None, 7, 7, 2048)
output: (None, 7, 7, 2048)
pose_avg_pool: AveragePooling2D
input: (None, 7, 7, 2048)
output: (None, 1, 1, 2048)
pose_flatten: Flatten
input: (None, 1, 1, 2048)
output: (None, 2048)
pose_output_layer: Dense
input: (None, 2048)
output: (None, 26)
average_1: Average
input: [(None, 26), (None, 26)]
output: (None, 26)

Appendix

Poster



UNIVERSITI TUNKU ABDUL RAHMAN

Human pose stream for multi-stream convolutional network in Video Action Classification

Supervisor: Dr. Tan Hung Khoon

Jackson Tan Zhe Sheng 1402698CS

Abstract

Multi-stream CNN have obtained great achievement for video processing tasks such as the human action classification. In this work, we proposed a multi-stream architecture which combined the spatial stream (still video frames) with a new modality of human pose. The input of the human pose stream consists of multiple human pose frames without the background (noises reduction) that stacked together which computed through a deep human posture estimation model (OpenPose). Our pilot studies show that the improvement of the performance in terms of accuracy by fusing the spatial stream and human pose stream (91.3%) which out-perform either the spatial (87.1%) or human pose stream (71.0%) alone on the UCF-101 dataset (26 classes).

Project Objective

- The proposed system from this project aim to recognize and classify the human actions within the input video and propose a new modality for the multi-stream CNN.

Training, validation accuracy and loss

Data-Preprossing

Original Frame
Horizontal Flipped Top Left Cropping
Horizontal Flipped Top Right Cropping
Horizontal Flipped Bottom Left Cropping
Horizontal Flipped Bottom Right Cropping
Horizontal Flipped Center Cropping
Stacked OpenPose Frame
Top Left Cropping
Top Right Cropping
Bottom Left Cropping
Bottom Right Cropping
Center Cropping

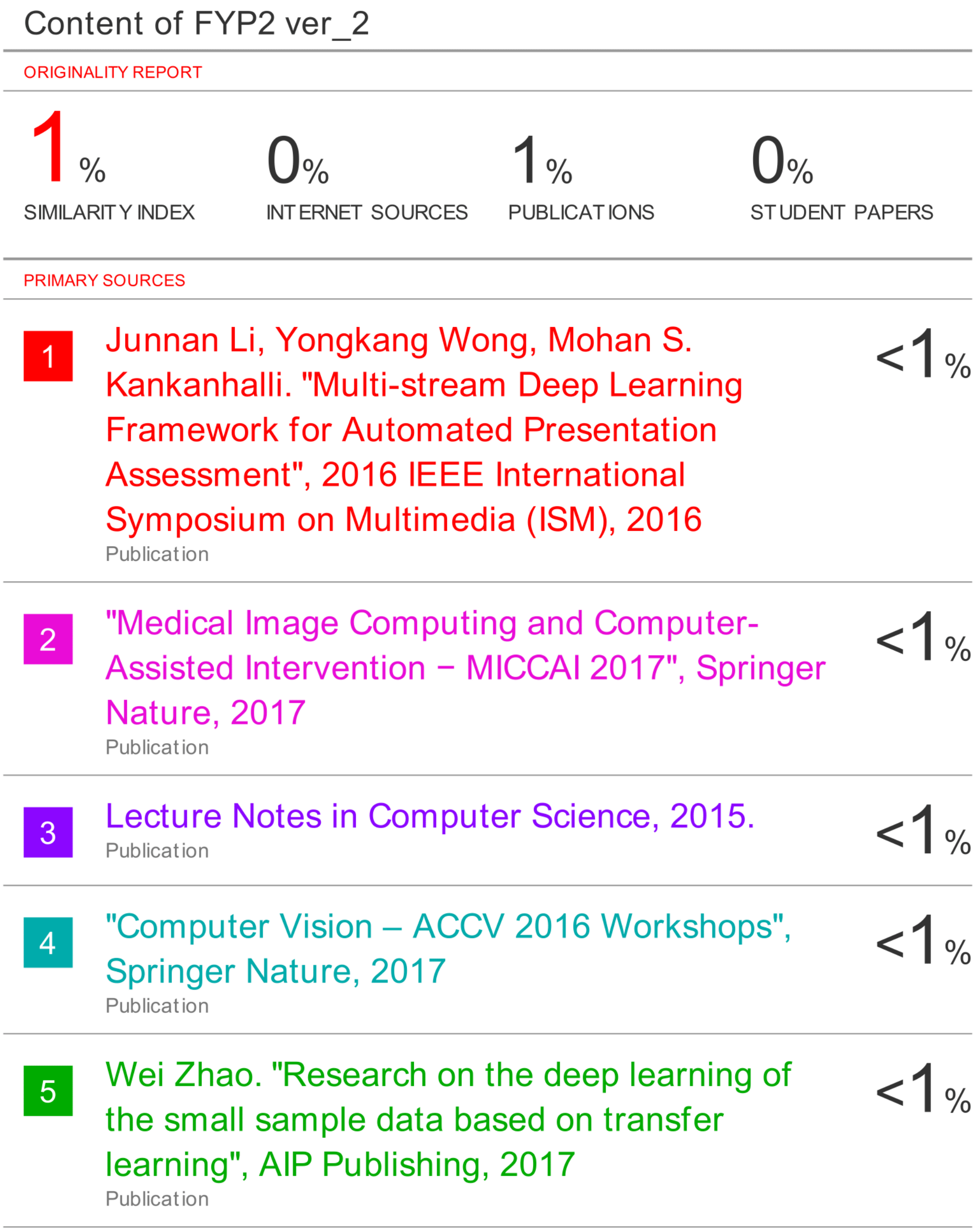
Multi-stream archicture overview

Conclusion

Our proposed multi-stream architecture with the spatial stream combined with a new modality of the human pose by average fusion able to achieve better result than the spatial stream alone while the human pose stream alone is not beneficial due to the loss of the spatial information is critical. However, when both of the static visual information and a series of human pose information are considered together, it able to performed better classification than either stream alone which proved that the importance of the multi-modalities processing within the video.

Appendix

Turnitin Similarity Report



Content of FYP2 ver_2

ORIGINALITY REPORT

1%	0%	1%	0%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Junnan Li, Yongkang Wong, Mohan S. Kankanhalli. "Multi-stream Deep Learning Framework for Automated Presentation Assessment", 2016 IEEE International Symposium on Multimedia (ISM), 2016 Publication	<1%
2	"Medical Image Computing and Computer-Assisted Intervention − MICCAI 2017", Springer Nature, 2017 Publication	<1%
3	Lecture Notes in Computer Science, 2015. Publication	<1%
4	"Computer Vision – ACCV 2016 Workshops", Springer Nature, 2017 Publication	<1%
5	Wei Zhao. "Research on the deep learning of the small sample data based on transfer learning", AIP Publishing, 2017 Publication	<1%

Appendix

Universiti Tunku Abdul Rahman			
Form Title : Supervisor's Comments on Originality Report Generated by Turnitin for Submission of Final Year Project Report (for Undergraduate Programmes)			
Form Number: FM-IAD-005	Rev No.: 0	Effective Date: 01/10/2013	Page No.: 1of 1

FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY

Full Name(s) of Candidate(s)	
ID Number(s)	
Programme / Course	
Title of Final Year Project	

Similarity	**Supervisor's Comments** **(Compulsory if parameters of originality exceeds the limits approved by UTAR)**
Overall similarity index:________ % **Similarity by source** Internet Sources: _____________% Publications: _____________% Student Papers: _____________%	
Number of individual sources listed of more than 3% similarity: _________	
Parameters of originality required and limits approved by UTAR are as Follows: **(i) Overall similarity index is 20% and below, and** **(ii) Matching of individual sources listed must be less than 3% each, and** **(iii) Matching texts in continuous block must not exceed 8 words** ***Note: Parameters (i) – (ii) shall exclude quotes, bibliography and text matches which are less than 8 words.***	

Note Supervisor/Candidate(s) is/are required to provide softcopy of full set of the originality report to Faculty/Institute

Based on the above results, I hereby declare that I am satisfied with the originality of the Final Year Project Report submitted by my student(s) as named above.

______________________________	______________________________
Signature of Supervisor	Signature of Co-Supervisor
Name: __________________________	Name: __________________________
Date: ___________________________	Date: ___________________________

Appendix

UNIVERSITI TUNKU ABDUL RAHMAN

FACULTY OF INFORMATION & COMMUNICATION TECHNOLOGY (PERAK CAMPUS)

CHECKLIST FOR FYP2 THESIS SUBMISSION

Student Id	
Student Name	
Supervisor Name	

TICK (√)	**DOCUMENT ITEMS** Your report must include all the items below. Put a tick on the left column after you have checked your report with respect to the corresponding item.
	Front Cover
	Signed Report Status Declaration Form
	Title Page
	Signed form of the Declaration of Originality
	Acknowledgement
	Abstract
	Table of Contents
	List of Figures (if applicable)
	List of Tables (if applicable)
	List of Symbols (if applicable)
	List of Abbreviations (if applicable)
	Chapters / Content
	Bibliography (or References)
	All references in bibliography are cited in the thesis, especially in the chapter of literature review
	Appendices (if applicable)
	Poster
	Signed Turnitin Report (Plagiarism Check Result - Form Number: FM-IAD-005)

I, the author, have checked and confirmed all the items listed in the table are included in my report. ________________________ (Signature of Student) Date:	Supervisor verification. Report with incorrect format can get 5 mark (1 grade) reduction. ________________________ (Signature of Supervisor) Date: