Digital Architecture for Real-Time CNN-based Face Detection for Video Processing

Smrity Bhattarai [1], Arjuna Madanayake [1]
Renato J. Cintra [2]
Stefan Duffner [3], Christophe Garcia [3]

Department of Electrical and Computer Engineering,
University of Akron, Akron, OH, USA [1]

Departamento de Estatística
UFPE, Recife, Brazil [2]

LIRIS, INSA
Lyon, France [3]

June 27, 2017

Outline

Image Processing and Face Detection

- The method of converting an image into a digital form to get an enhanced image or extract useful information by performing some operations is called image processing
- Processing an image signal using mathematical operations for which an input can be either image, video or graph and the output is either image or set of characteristics of the corresponding image
- Image processing includes applying different sets of signal processing methods on images by treating them mostly as two dimensional (2D) signals

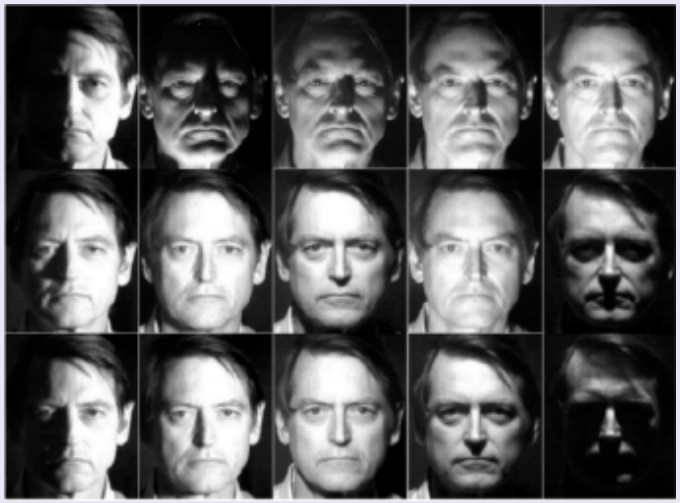
Applications of Image Processing

1. Face Detection and Recognition
2. Defense Surveillance
3. Moving Object Tracking
4. Content based Image Retrieval
5. Image and Video Compression
6. Image Sharpening and Restoration

- Explored face detection technique that uses concept of feed forward neural network which is one of the most fundamental image processing technique

Challenges in Face detection

Illumination Variation



Expression Variation

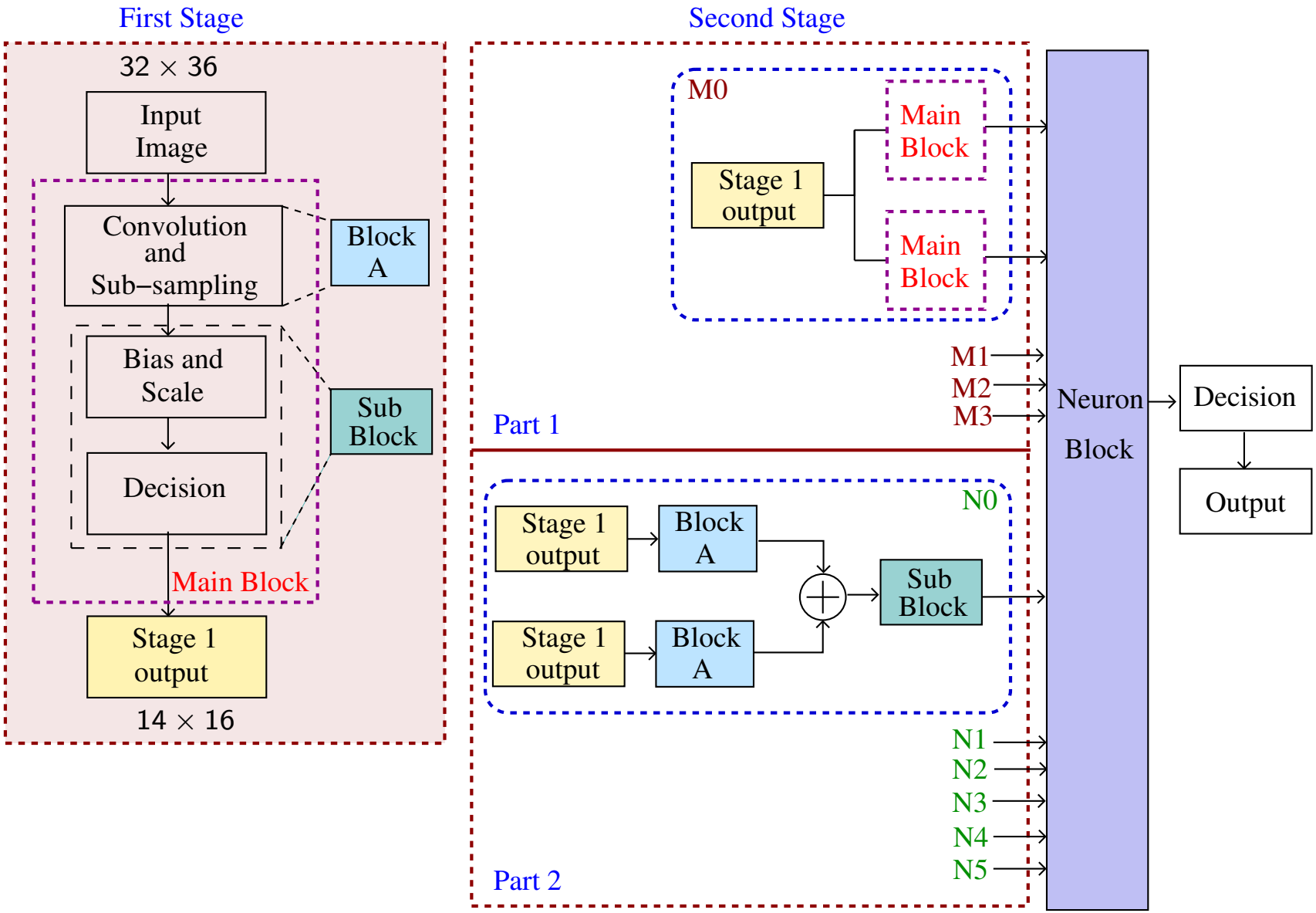
Variation in Image Position and Scale

Variation in Quality of Image

Proposed Approach

- Explore a digital architecture for real-time face detection and recognition system
- Approximate hardware realization scheme learning algorithm of convolutional neural networks (CNN) [1] to improve the efficiency of the hardware implementation of neural networks and to reduce the hardware complexity
- Evaluate accuracy of the system under different constraints
- Analyze hardware complexity of the system

Block Diagram of the Proposed Approach[1]



Proposed Approach - First Stage

- Images are matrices of order 32×36
- SS1 is sub-sampling operation
- K1, K2, K3, K4 are four kernels of order 5×5
- b_{11}, b_{12}, b_{13}, b_{14} are biases
- α_1,α_2,α_3,α_4 are scale factors
- D is the decision block
- D_1, D_2, D_3, D_4 are four output images after first stage and are of order 14×16

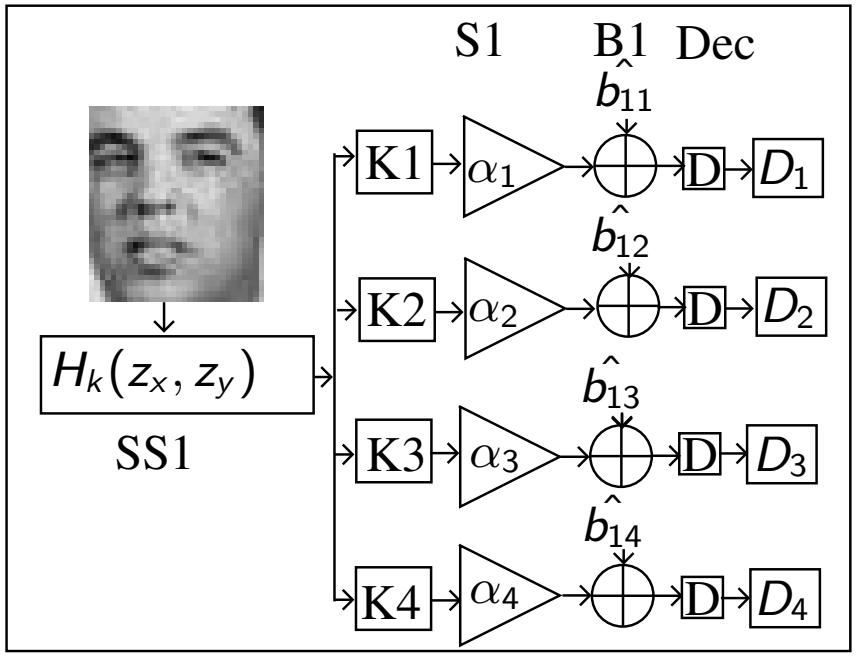


Figure: First Stage

Proposed Approach Second Stage

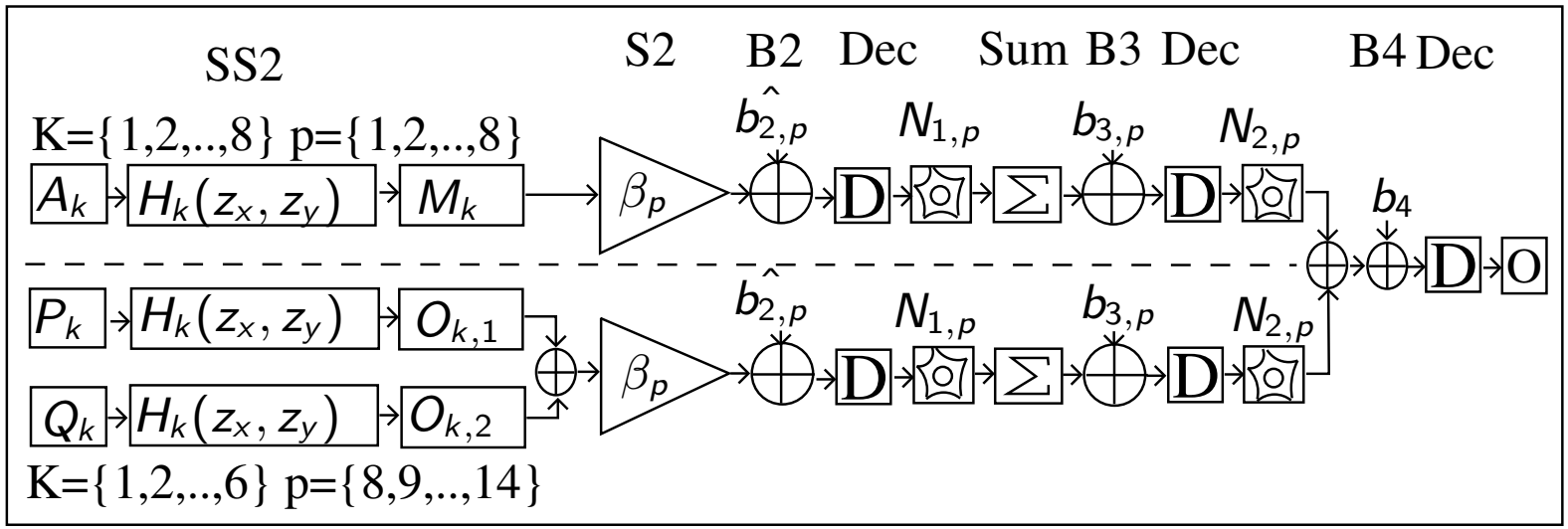


Figure: Second Stage

- Images are of matrices of order 14×16
- M_k, $O_{k,1}$ and $O_{k,2}$ are kernels of order 3×3
- $\hat{b}_{2,p}$, $b_{3,p}$ and b_4 are biases
- β_p and D are scale factors and decision block respectively
- $N_{1,p}$ are neurons of order 6×7
- $N_{2,p}$ is a single neuron

Hardware Implementation

Convolution and Sub-sampling

$$J(i_k, j_k) = \sum_{m=1}^{M} \sum_{n=1}^{N} K(m,n).I[(i_k - 1) + m, (j_k - 1) + n] \quad (1)$$

$$H_k(z_x, z_y) = \frac{(1 + z_x^{-1})(1 + z_y^{-1})}{4} \quad (2)$$

where,

J is the convolution operation

H is the sub-sampling operation

K is kernel matrix of order $m \times n$

I is image matrix

Convolution and Sub-sampling

From Equations 1 and 2

$$Q_C = \sum_{i=L}^{L+1} \sum_{j=S}^{S+1} x(i,j)$$

where,
$L = 2(T \backslash A) + (C-1) \backslash R + 1$
$S = 2[(T-1)\%A] + (C-1)\%R + 1$
T = Time Stamp
R = Order of Kernel = 5
C = Column Number

$$Q_1 = x_{11} + x_{12} + x_{21} + x_{22}$$
$$Q_{25} = x_{55} + x_{56} + x_{65} + x_{66}$$

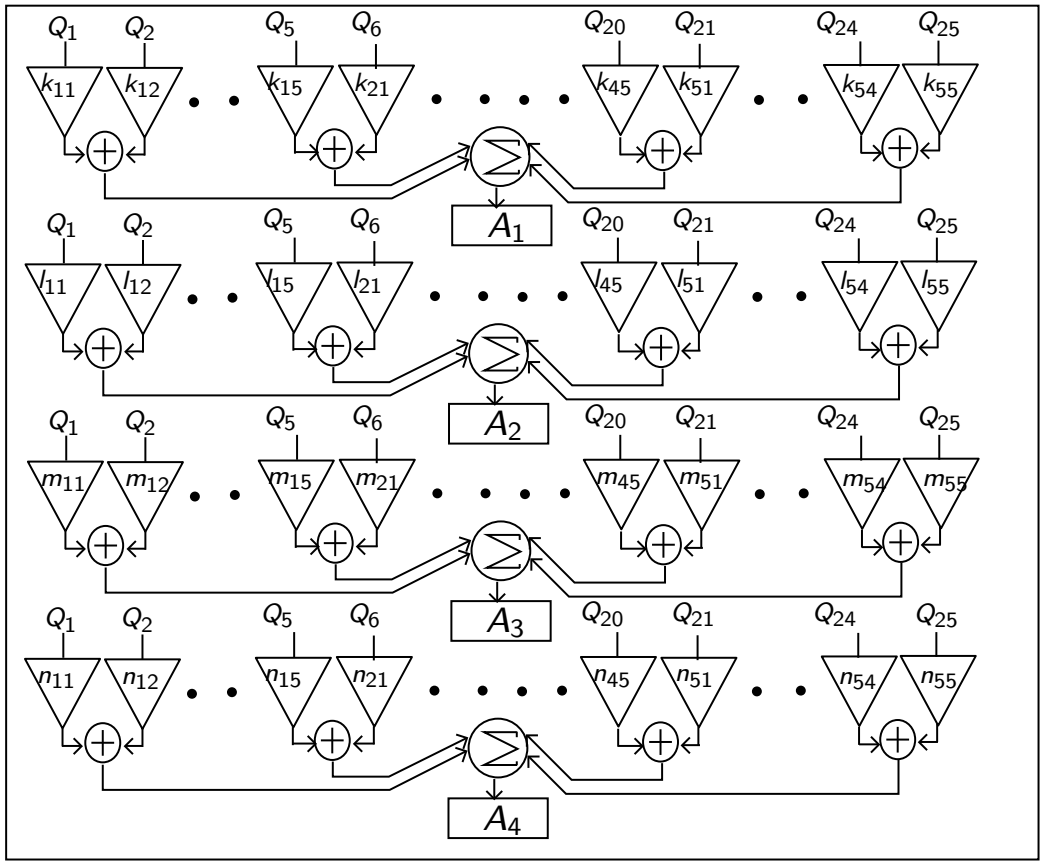


Figure: Combined Convolution and Sub-sampling Block

Scaling and Bias Block

$S1 = (\alpha_1, \alpha_2, \alpha_3, \alpha_4)$

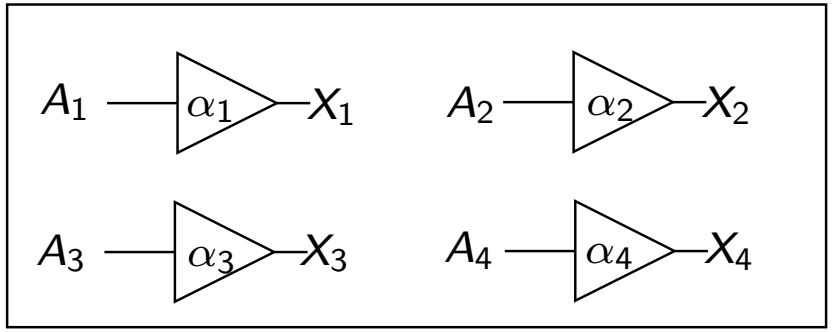


Figure: Scaling Block

$B1 = \hat{b_{1,k}}$

$\hat{b_{1,k}} = b_{1,k}.\alpha_k + b_{1,2,k}$

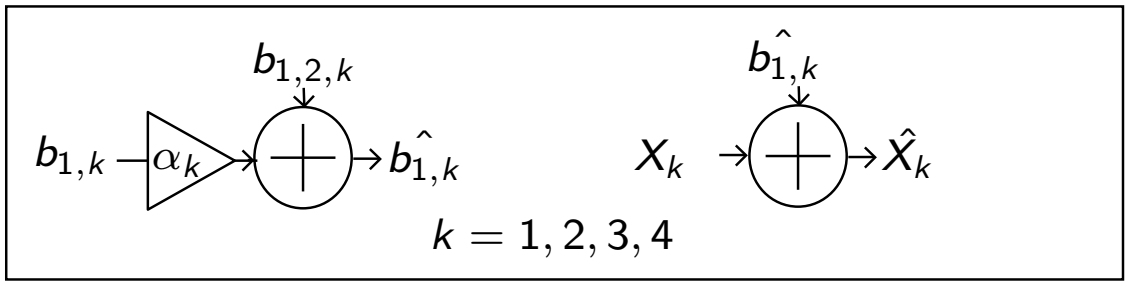


Figure: Bias Block

Decision Block

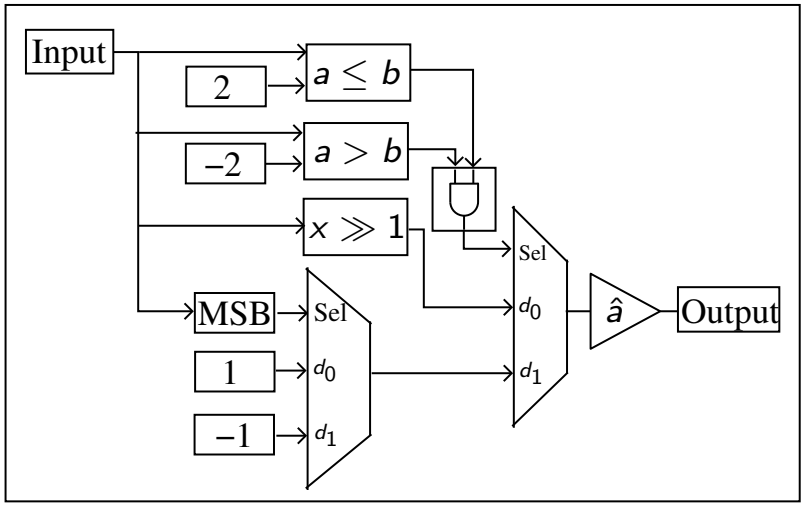


Figure: Decision Block

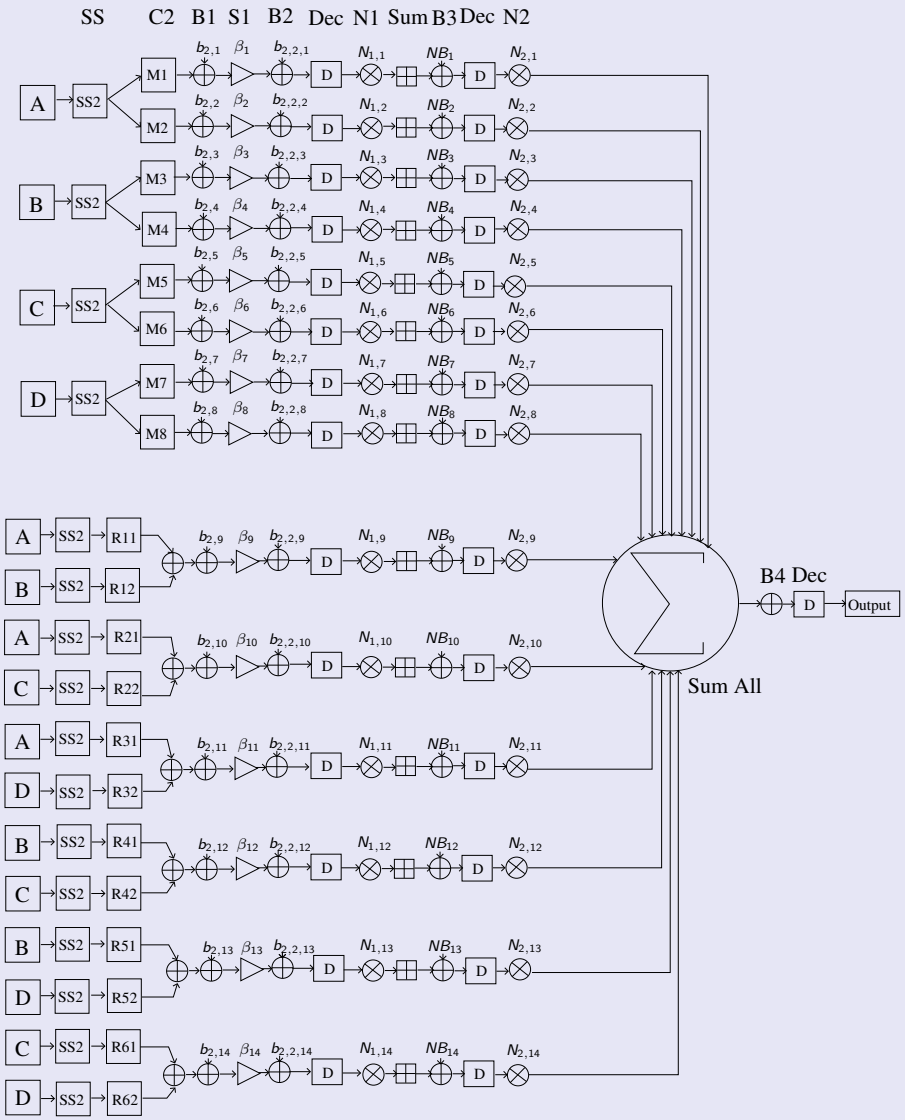
The decision block is basically linear-2 approximation[2] and given as:

$$\sigma(x) = \hat{a}. \begin{cases} -1 & \text{if } x < -2 \\ x/2 & \text{if } -2 \leq x < 2 \\ 1 & \text{if } x \geq 2 \end{cases}$$

where, $\hat{a} = 7/4$

Hardware Implementation

Detailed Block Diagram of Second Stage



Convolution and Sub-sampling

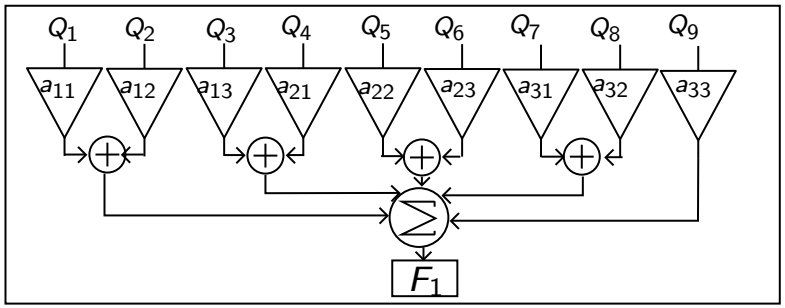


Figure: 1st part of 2nd Stage of Combined Convolution Sub-sampling Block

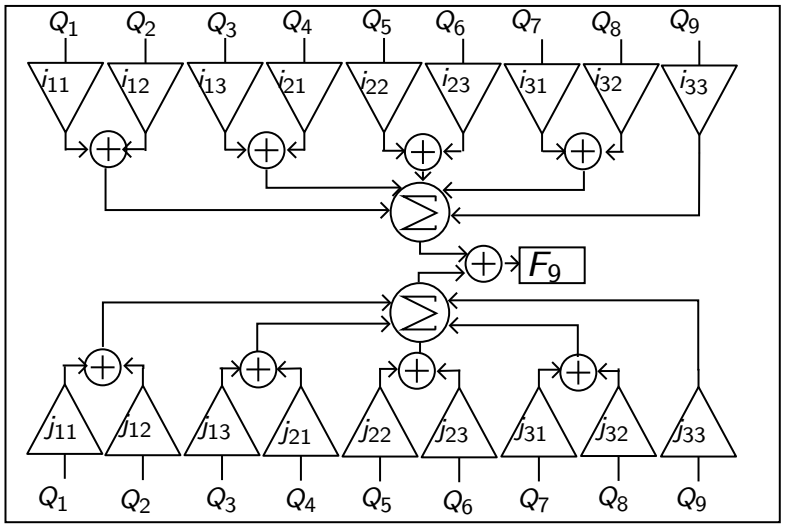


Figure: 2nd part of 2nd Stage of Combined Convolution Sub-sampling Block

- Followed by a scale, bias and decision block like in first stage

Neuron Block including Bias and Decision Block

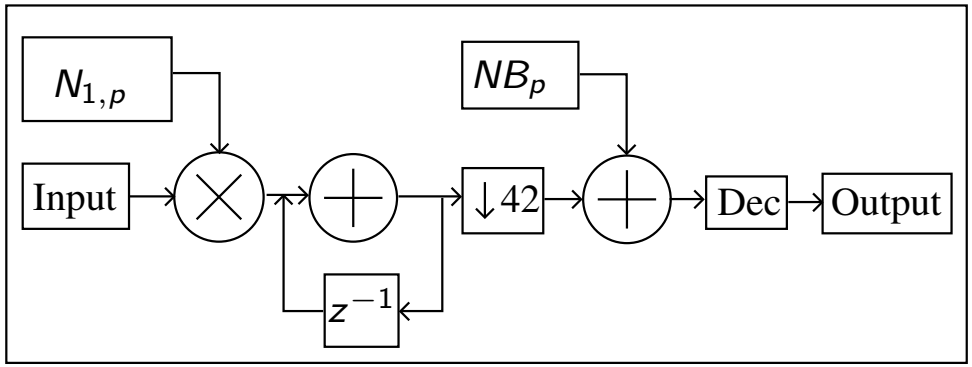


Figure: Neuron Block

- Total 14 matrices of size 6×7 and is represented by N_p where $p = 1, 2, ..14$
- Contains bias for each neuron block represented by the set NB_p

$$NB_p = \hat{b_{3,p}}$$

- Followed by an activation function represented by decision block
- Scalar multiplied with the matrix generated after second stage of decision block element by element and passes through the added decision block

Neuron Multiplication Block with Final bias and Decision Block

- $\hat{J}_p$ is the output of Neuron Block
- The set $\hat{J}_p$ is multiplied with corresponding neurons set called '$N_{2,p}$'
- All the elements of the resulting set $\hat{V}_p$ is added together to get a value 'W'
- The value W obtained after Neuron Multiplication is biased with bias B4
- The output is then passed through the decision block giving rise to obtain final output

Final Output

- If the final output is positive, the image taken at the beginning is a face else the image taken is not a face

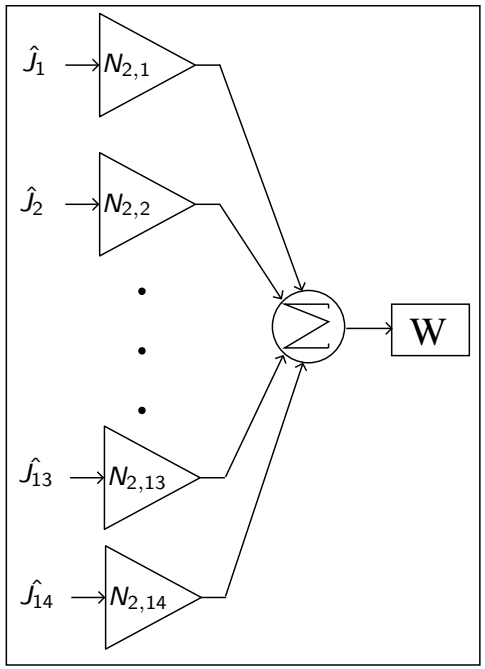


Figure: Neuron Multiplication Block

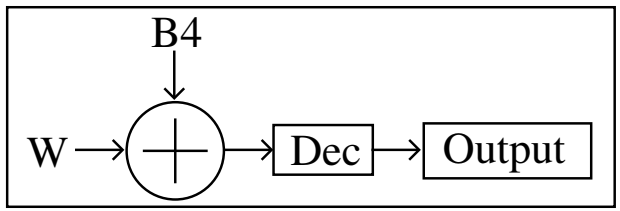


Figure: Final Bias and Decision Block

Analysis and Results

- Tests were carried out using different set of images
- All the input images were standardized to size 32×36 and in .pgm (Portable Gray Map) format
- The algorithm was first modelled and tested in MATLAB Simulink using Xilinx block set
- Tested on Xilinx Virtex-6 ML605 FPGA Evaluation Kit
- 1164 images and were processed

Figure: Sample test images

Accuracy Details

MATLAB Accuracy Details

Tested Images	1164
MATLAB Correctly Detected	1124
Accuracy of MATLAB	96.56%

Change in Accuracy of Design After Changing the Word-Length of Scale and Biases

Word Length			**Correctly Detected Xilinx**	**Accuracy**
Part	**Scale**	**Bias**		
Integer	8	8	1123	96.47%
Fractional	5	5		
Integer	7	7	1120	96.21%
Fractional	4	4		
Integer	6	6	1096	94.15%
Fractional	3	3		
Integer	5	5	1084	93.12%
Fractional	2	2		
Integer	4	4	567	48.71%
Fractional	1	1		

Hardware Complexity

Hardware Consumption Along with Timing Details

Scale and Bias		Stage	Hardware Consumption			
Integer	Fractional		LUTs	FFs	T_{cpd} (ns)	F_{max} (MHz)
8	5	1	3345	2770	3.17	315.25
		2	8327	8435	4.84	206.35
7	4	1	3335	2767	3.16	316.45
		2	8277	8413	4.76	210.03
6	3	1	3288	2765	3.12	320
		2	8144	8401	4.76	210.03
5	2	1	3278	2761	3.09	322.99
		2	7685	7909	4.47	223.31
4	1	1	3278	2758	3.04	328.29
		2	5496	5678	4.47	223.31

Hardware Complexity and Accuracy

Change in Accuracy After Changing the Word-Length of Kernel from the First stage

Word - Length		Correctly Detected Xilinx	Accuracy
Integer	Fractional		
10	5	1123	96.47%
9	4	1122	96.39%
8	3	1113	95.61%
7	2	1118	96.04%
6	1	1103	94.75%

Hardware Consumption Along with Timing Details

Kernel from first stage		Hardware Consumption			
Integer	Fractional	LUTs	FFs	T_{cpd} (ns)	F_{max} (MHz)
10	5	3351	2770	3.25	306.93
9	4	3160	2786	3.22	309.88
8	3	2974	2797	3.15	316.85
7	2	2826	2779	3.14	317.76

Hardware Complexity and Accuracy

Change in Accuracy After Changing the Word-Length of Kernel from the Second Stage

Word - Length		Correctly Detected Xilinx	Accuracy
Integer	Fractional		
8	5	1123	96.47%
7	4	1122	96.39%
6	3	1120	96.21%
5	2	1105	94.93%
4	1	1101	94.58%

Hardware Consumption Along with Timing Details

Kernel from second stage		Hardware Consumption			
Integer	Fractional	LUTs	FFs	T_{cpd} (ns)	F_{max} (MHz)
7	4	7918	8419	4.57	218.96
6	3	7492	8149	4.56	219.29
5	2	7187	7880	4.49	222.71
4	1	6944	7491	4.43	225.73

Hardware Complexity and Accuracy

Change in Accuracy After Changing the Word-Length of Neuron Biases

Word - Length		Correctly Detected Xilinx	Accuracy
Integer	Fractional		
7	5	1123	96.47%
6	4	1123	96.47%
5	3	1121	96.30%
4	2	1122	96.39%
3	1	1120	96.21%

Hardware Consumption Along with Timing Details

Neuron		Hardware Consumption			
Integer	Fractional	LUTs	FFs	T_{cpd} (ns)	F_{max} (MHz)
7	5	8149	8263	5.42	184.5
6	4	8104	8217	4.85	206.14
5	3	8063	8174	4.52	220.94
4	2	8015	8135	4.51	221.48
3	1	7998	8091	4.50	221.92

Conclusions

- A hardware system capable of face detection and recognition was designed
- The implementation paves a path for an efficient architecture designed to detect highly variable face patterns
- The system is a fully trained architecture comprising of a pipeline of convolution, sub-sampling and neural networks, such that no pre-processing of the image is required
- The experimental results show high accuracy ($\approx$96%)
- The proposed model can be a good solution in various facial recognition and detection application

References I

[1] C. Garcia and M. Delakis, "Convolutional face finder: A neural architecture for fast and robust face detection," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 26, no. 11, pp. 1408–1423, 2004.

[2] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Networks*, vol. 61, pp. 85–117, 2015.

Thank You