

Ho Chi Minh City University of Technology
Faculty of Computer Science and Engineering



Assignment
CONVOLUTION OPERATION
Course: Computer Architecture Lab
(CO2008)

Teacher: Nguyễn Thành Lộc
Class: CC10

Name	Student ID
Vũ Hải Tuấn	2353280

2024

Contents

<i>I. Overview.....</i>	<i>3</i>
<i>II. Key Components.....</i>	<i>3</i>
<i>III. Algorithm Implementation.....</i>	<i>3</i>
<i>IV. Mips Code</i>	<i>4</i>
<i>V. Images of test runs.....</i>	<i>5</i>
<i>VI. Flowchart</i>	<i>6</i>

I. Overview

This program implements a 2D convolution operation, a common technique in image processing and deep learning. It processes an input matrix (image) using a kernel matrix, applying specified padding and stride values.

II. Key Components

1. Data Structure

```
.data
    image:      .float 0:2500      # Input matrix storage
    kernel:     .float 0:2500      # Kernel matrix storage
    output:     .float 0:2500      # Result matrix storage
    padded:     .float 0:10000     # Padded input matrix
```

2. Main Parameters

N: Input matrix size (3–7)

M: Kernel size (2–4) p:

Padding size (0–4)

s: Stride value (1–3)

III. Algorithm Implementation

1. Input Processing

Reads matrix dimensions and parameters from an input file.

Validates the range of parameters.

Calculates output dimensions using the formula:

$$\text{output_size} = \frac{(N + 2p - M)}{s} + 1$$

2. Matrix Reading

Implements float parsing from text

Handles both integer and decimal parts

Supports negative numbers
Stores values in floating-point format

3. Convolution Operation

The convolution process follows these steps:

a) Padding Implementation

Creates padded matrix of size $(N + 2p) \times (N + 2p)$

Initializes padding with zeros

Copies original image to center of padded matrix

b) Convolution Computation

For each output position (i,j):

Initialize sum = 0

For each kernel position (k,l):

input_pos = stride * i + k

output_pos = stride * j + l

sum += padded[input_pos][output_pos] * kernel[k][l]

output[i][j] = sum

IV. Mips Code

1. Main Function (main):

Opens the file for reading.

Reads values for N, M, p, and s, and validates them based on the specified ranges.

Calculates the output size using the formula:

Reads the image and kernel matrices into memory.

Performs the convolution operation.

Displays the output matrix.

2. Reading and Validating Integers (readint):

Reads an integer value from the buffer (which stores the contents of the input file).

Skips non-numeric characters (such as spaces, newlines, and decimal points).

Checks the validity of the values for N, M, p, and s by ensuring they fall within specified ranges.

3. Reading Float Matrices (read_float_matrix):

Reads floating-point values from the buffer and stores them in the image or kernel arrays.

Each value is processed by converting from ASCII to integer and handling the decimal part.

The result is then stored in the floating-point registers and saved into memory.

4.Convolution:

The convolution process would typically involve sliding the kernel over the image and performing element-wise multiplication followed by summation.

5.Display Matrix (display_matrix):

Displays the contents of the output matrix.

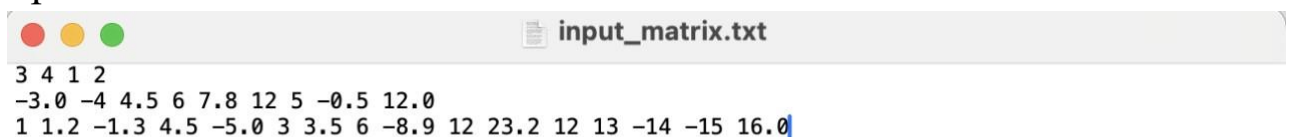
The values are printed with a space separator and are followed by a newline after each row.

6.Create file output_matrix.txt.

V.Images of test runs

1.Test1

Input file :



```
3 4 1 2
-3.0 -4 4.5 6 7.8 12 5 -0.5 12.0
1 1.2 -1.3 4.5 -5.0 3 3.5 6 -8.9 12 23.2 12 13 -14 -15 16.0
```

Output file:



```
530.4600
```

2.Test2

Input file:

```
input_matrix.txt
4 3 3 3
1 1.2 -1.3 4.5 -5.0 3 3.5 6 -8.9 12 23.2 12 13 -14 -15 16.0
-3.0 -4 4.5 6 7.8 12 5 -0.5 12.0
```

Output file:

```
output_matrix.txt
0.0000 0.0000 0.0000 0.0000 249.6500 82.5000 0.0000 -50.5000 -48.0000
```

VI.Flowchart

