

Cairo University

Faculty of Engineering

Computer Engineering Department



Cairo University

LogIm

CMP2030

Team Members Info:

Name	Sec	BN	Email
Sarah Mohamed Hossam	1	30	sarahelzayat@outlook.com
Ahmed Atta Abdallah	1	6	aatta1082001@gmail.com
Fady Adel	2	6	fady.adel2001@gmail.com
Nour Ziad Almulhem	2	29	nouralmulhem@gmail.com

Used algorithms:

1. Quine mcCluskey algorithm to minimize logical expressions
2. Find Contours
3. Hog for feature extraction
4. SVC classification
5. SGD classification

Experiment results and analysis:

- Level of variety used in test cases:
Images of different orientations, lighting, sizes and skewness were used to ensure a decent coverage.
- Performance and accuracy:
 - Letter extraction accuracy:
The module extracts letters from expressions and tables.

Test image:

A B C D E F = ~ () →

Results:

A B C D E F = ~ () →

- Classification accuracy:
Using the SVC and SGD classifiers from the SVM library with the HOG for feature extraction, the accuracy of character classification is great.
There are 3 types of inputs to be classified, 0's and 1's, letters only (A, B, C, D, E, F), letters and symbols (A, B, C, D, E, F, +, (,), ~, →), so 3 models trained of different datasets were used to ensure accuracy.
However, the classifier couldn't differentiate between E and F with the current feature description, so another layer of classification was added to differentiate between them.
Still, with an unusual font/handwriting, it fails to predict the right results.

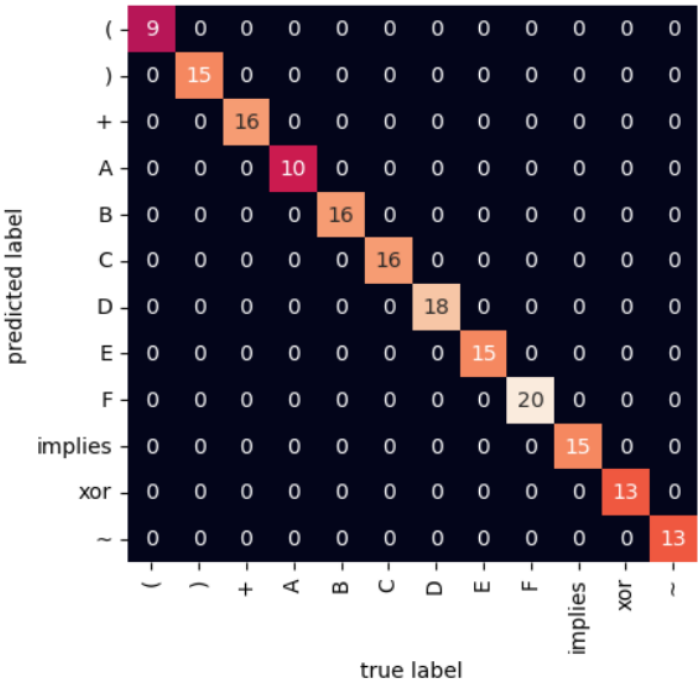
Test image:

A B C D E F → + ⊕ ~ ()

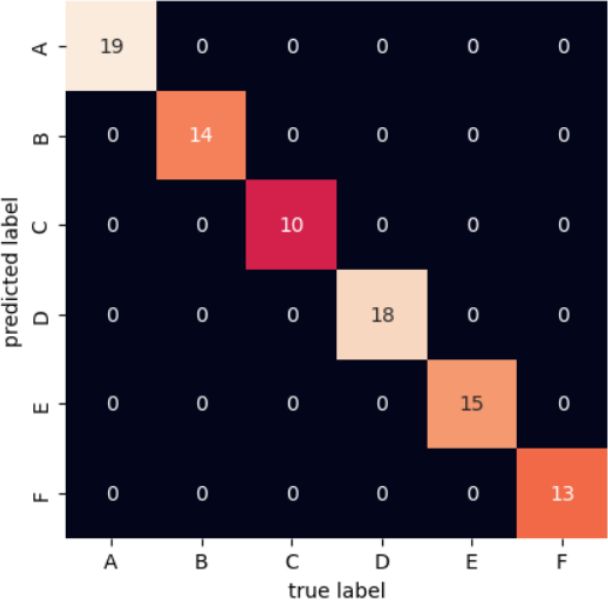
Results:

```
['A' 'B' 'C' 'D' 'E' 'F' 'implies' '+' 'xor' '~' '(' ')']
```

Letters and symbols classifier confusion matrix:



Letters only classifier



predicted label	E	15	0
	F	0	15
		E	F
		true label	

The above results were based on handwritten datasets.

- Preprocessing accuracy:
trapezoidal detector module can detect the rectangular scanned paper when 4 corners appear clearly otherwise it gets the maximum area inside the paper whether it was the table in expression generation or the expression itself in case of truth table generation
- Table detection accuracy:
The module mainly depends on detecting the vertical and horizontal lines of the table so it will result in wrong detection if table is a little skewed which is already handled in preprocessing step

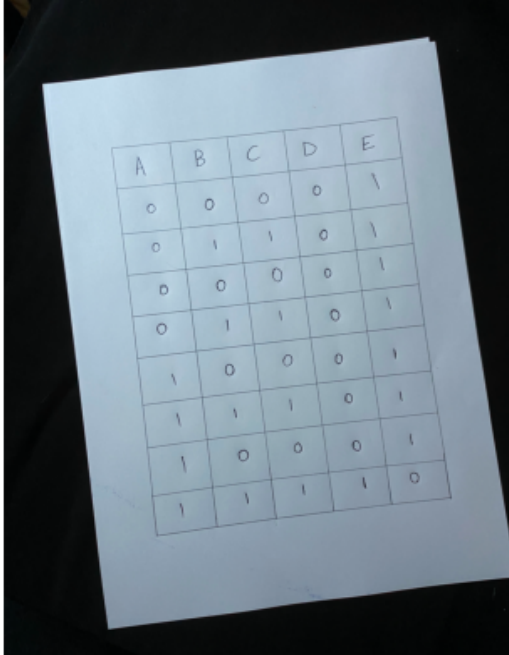
Work Division:

Team Member	Work load
Ahmed Ata	Letters extraction in each Classification Expression to truth table
Sarah Elzayat	Expression preprocessing Counting and Extracting rows of expressions Classification
Fady Adel	Table preprocessing Table Detection and Cell Extraction
Nour Ziad	Table preprocessing Table Detection and Cell Extraction

TEST CASES

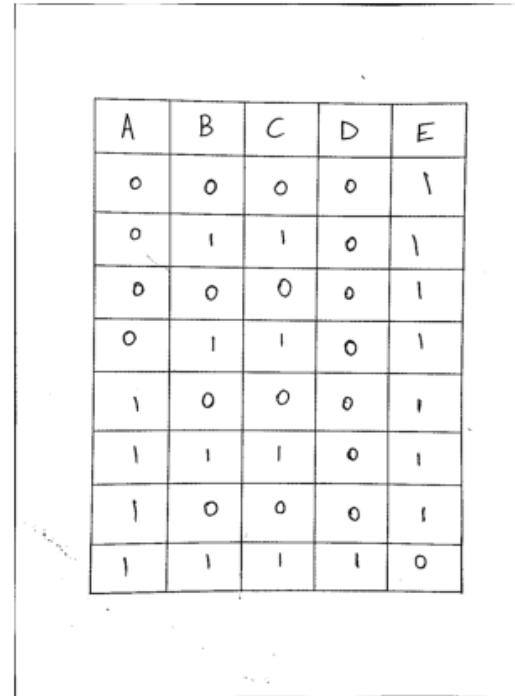
Preprocessing:

Input image



A	B	C	D	E
0	0	0	0	1
0	1	1	0	1
0	0	0	0	1
0	1	1	0	1
1	0	0	0	1
1	1	1	0	1
1	0	0	0	1
1	1	1	1	0

Result



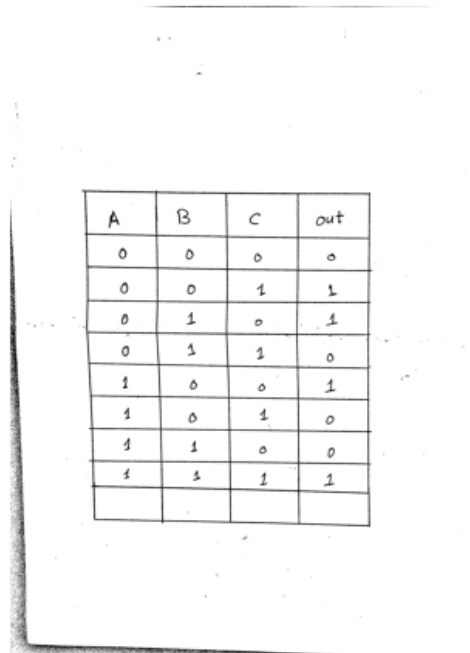
A	B	C	D	E
0	0	0	0	1
0	1	1	0	1
0	0	0	0	1
0	1	1	0	1
1	0	0	0	1
1	1	1	0	1
1	0	0	0	1
1	1	1	1	0

Input image



A	B	C	out
0	0	0	0
0	0	1	1
0	1	0	1
1	0	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

Result



A	B	C	out
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

Table detection:

input image

A	B	C	D	E
0	0	0	0	1
0	1	1	0	1
0	0	0	0	1
0	1	1	0	1
1	0	0	0	1
1	1	1	0	1
1	0	0	0	1
1	1	1	1	0

Row 1

(1) A	(2) B	(3) C	(4) D	(5) E
-------	-------	-------	-------	-------

Row 2

(1) 0	(2) 0	(3) 0	(4) 0	(5) 1
-------	-------	-------	-------	-------

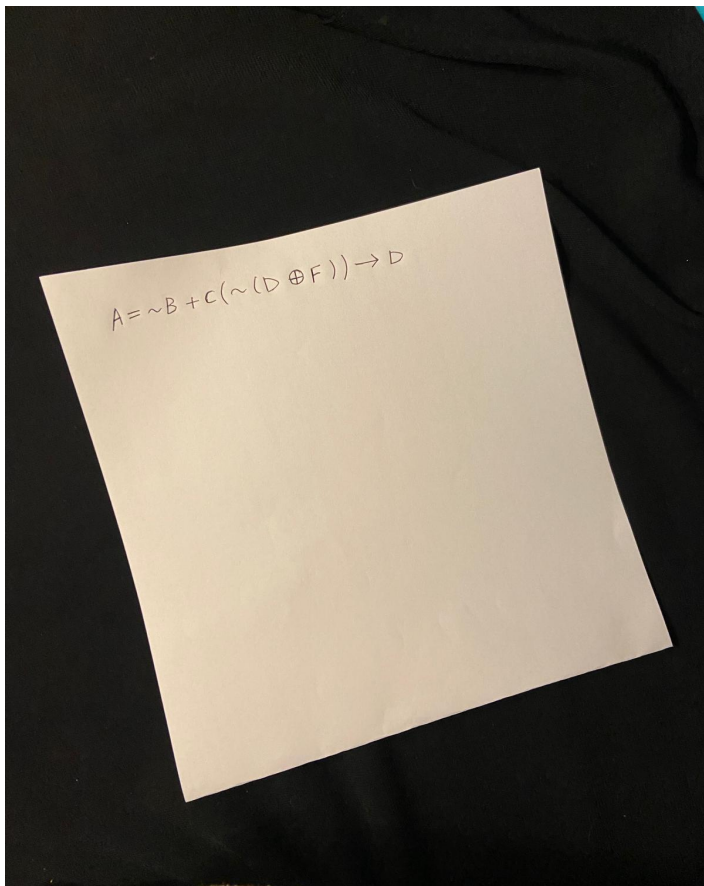
Row 3

(1) 0	(2) 1	(3) 1	(4) 0	(5) 1
-------	-------	-------	-------	-------

Row 4

(1) 0	(2) 0	(3) 0	(4) 0	(5) 1
-------	-------	-------	-------	-------

ExpressionsToTT:



B	F	D	C	$\sim B$ or C and $(\sim(D \text{ xor } F))$ implies D
1	1	1	1	1
1	1	1	0	1
1	1	0	1	1
1	1	0	0	1
1	0	1	1	1
1	0	1	0	1
1	0	0	1	0
1	0	0	0	1
0	1	1	1	1
0	1	1	0	1
0	1	0	1	0
0	1	0	0	0
0	0	1	1	1
0	0	1	0	1
0	0	0	1	0
0	0	0	0	0

$A = B \oplus C$
 $A = B \rightarrow C$
 $A = BCDEF$
 $A = (B+C) + \sim(D \rightarrow E)$
 $B = \sim A$

B	C	B xor C
1	1	0
1	0	1
0	1	1
0	0	0

B	C	B implies C
1	1	1
1	0	0
0	1	1
0	0	1

E	F	D	B	C	B and C and D and E and F
1	1	1	1	1	1
1	1	1	1	0	0
1	1	1	0	1	0
1	1	1	0	0	0
1	1	0	1	1	0
1	1	0	1	0	0
1	1	0	0	1	0
1	1	0	0	0	0
1	0	1	1	1	0
1	0	1	1	0	0
1	0	1	0	1	0
1	0	1	0	0	0
1	0	0	1	1	0
1	0	0	1	0	0
1	0	0	0	1	0
1	0	0	0	0	0
0	1	1	1	1	0
0	1	1	1	0	0
0	1	1	0	1	0
0	1	1	0	0	0
0	1	0	1	1	0
0	1	0	1	0	0
0	1	0	0	1	0
0	1	0	0	0	0
0	0	1	1	1	0
0	0	1	1	0	0
0	0	1	0	1	0
0	0	1	0	0	0
0	0	0	1	1	0
0	0	0	1	0	0
0	0	0	0	1	0
0	0	0	0	0	0

B	E	D	C	(B or C) or ~ (D implies E)
1	1	1	1	1
1	1	1	0	1
1	1	0	1	1
1	1	0	0	1
1	0	1	1	1
1	0	1	0	1
1	0	0	1	1
1	0	0	0	1
0	1	1	1	1
0	1	1	0	0
0	1	0	1	1
0	1	0	0	0
0	0	1	1	1
0	0	1	0	1
0	0	0	1	1
0	0	0	0	0

A	~ A
1	0
0	1

$$A = \sim B + C(D \oplus E) + F$$

$$B = D + C + A$$

$\neg B \text{ or } C \text{ and } (D \text{ xor } E) \text{ or } F$

D or C or A

C	E	B	F	D	$\neg B \text{ or } C \text{ and } (D \text{ xor } E) \text{ or } F$
1	1	1	1	1	1
1	1	1	1	0	1
1	1	1	0	1	0
1	1	1	0	0	1
1	1	0	1	1	1
1	1	0	1	0	1
1	1	0	0	1	1
1	1	0	0	0	1
1	0	1	1	1	1
1	0	1	1	0	1
1	0	1	0	1	1
1	0	1	0	0	0
1	0	0	1	1	1
1	0	0	1	0	1
1	0	0	0	1	1
1	0	0	0	0	1
0	1	1	1	1	1
0	1	1	1	0	1
0	1	1	0	1	0
0	1	1	0	0	0
0	1	0	1	1	1
0	1	0	1	0	1
0	1	0	0	1	1
0	1	0	0	0	1
0	0	1	1	1	1
0	0	1	1	0	1
0	0	1	0	1	0
0	0	1	0	0	0
0	0	0	1	1	1
0	0	0	1	0	1
0	0	0	0	1	1
0	0	0	0	0	1

C	A	D	D or C or A
1	1	1	1
1	1	0	1
1	0	1	1
1	0	0	1
0	1	1	1
0	1	0	1
0	0	1	1
0	0	0	0

Results:

Table To expression:

A	B	C	D	E	F
0	0	0	0	0	1
0	0	1	0	1	0
0	1	0	0	1	0
0	1	1	0	1	0
1	0	0	0	1	0
1	0	1	0	1	0
1	1	0	0	1	0
1	1	1	1	1	0

rows numbers 9

cols numbers 6

num_outputs: 3

F0 = abc +

F1 = c + b + a +

F2 = a'b'c' +

A	B	C	D	E
0	0	0	0	1
0	0	0	1	1
0	0	1	0	1
0	0	1	1	1
0	0	1	0	0
0	1	0	1	1
0	1	0	1	1
0	1	1	0	1
0	1	1	1	1
1	0	0	0	1
1	0	0	1	1
1	0	1	0	1
1	1	0	1	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	0
1	1	1	1	1

```
rows numbers 17
cols numbers 5
num_outputs: 1
F0 = a' + b' + c' + d'
```

A	B	C	D	E	F
0	0	0	0	0	0
0	0	0	0	1	0
0	0	0	1	0	0
0	0	0	1	1	0
0	0	1	0	0	0
0	0	1	0	1	0
0	0	1	1	0	0
0	0	1	1	1	0
0	1	0	0	0	0
0	1	0	0	1	0
0	1	0	1	0	0
0	1	0	1	1	0
0	1	1	0	0	0
0	1	1	0	1	0
0	1	1	1	0	0
0	1	1	1	1	0
1	0	0	0	0	0
1	0	0	0	1	0
1	0	0	1	0	0
1	0	0	1	1	0
1	0	1	0	0	0
1	0	1	0	1	0
1	0	1	1	0	0
1	0	1	1	1	0
1	1	0	0	0	0
1	1	0	0	1	0
1	1	0	1	0	0
1	1	1	0	0	0
1	1	1	0	1	0
1	1	1	1	0	0
1	1	1	1	1	1

```
rows numbers 33
cols numbers 6
num_outputs: 1
F0 = abcde +
```

A	B	C	D	E	F
0	0	0	0	0	1
0	0	0	0	1	1
0	0	0	1	0	1
0	0	0	1	1	1
0	0	1	0	0	1
0	0	1	0	1	1
0	0	1	1	0	1
0	0	1	1	1	1
0	1	0	0	0	1
0	1	0	0	1	1
0	1	0	1	0	1
0	1	0	1	1	1
0	1	1	0	1	1
0	1	1	0	1	1
0	1	1	1	0	1
0	1	1	1	1	1
1	0	0	0	0	1
1	0	0	0	1	1
1	0	0	1	0	1
1	0	1	0	1	1
1	0	1	1	0	1
1	0	1	1	1	1
1	1	0	0	0	1
1	1	0	0	1	1
1	1	0	1	0	1
1	1	0	1	1	1
1	1	1	0	0	1
1	1	1	0	1	1
1	1	1	1	0	1
1	1	1	1	1	0

```
rows numbers 33
cols numbers 6
num_outputs: 1
F0 = a' + b' + c' + d' + e'
```

A	B	C	D	E	F
0	0	0	0	0	1
0	0	0	0	1	1
0	0	0	1	0	1
0	0	0	1	1	1
0	0	1	0	0	1
0	0	1	0	1	1
0	0	1	1	0	1
0	0	1	1	1	1
0	1	0	0	0	1
0	1	0	0	1	1
0	1	0	1	0	1
0	1	0	1	1	1
0	1	1	0	1	1
0	1	1	0	1	1
0	1	1	1	0	1
0	1	1	1	1	1
1	0	0	0	0	1
1	0	0	0	1	1
1	0	0	1	0	1
1	0	0	1	1	1
1	0	1	0	0	1
1	0	1	0	1	1
1	0	1	1	0	1
1	0	1	1	1	1
1	1	0	0	0	1
1	1	0	0	1	1
1	1	0	1	0	1
1	1	0	1	1	1
1	1	1	0	0	1
1	1	1	0	1	1
1	1	1	1	0	1
1	1	1	1	1	0

```
rows numbers 33
cols numbers 6
num_outputs: 1
F0 = a' + b' + c' + d' + e'
```


A	B	C	D	E	F
0	0	0	0	0	0
0	0	0	0	1	0
0	0	0	1	0	0
0	0	0	1	1	0
0	0	1	0	0	0
0	0	1	0	1	0
0	0	1	1	0	0
0	0	1	1	1	0
0	1	0	0	0	0
0	1	0	0	1	0
0	1	0	1	0	0
0	1	0	1	1	0
0	1	1	0	0	0
0	1	1	0	1	0
0	1	1	1	0	0
0	1	1	1	1	0
1	0	0	0	0	0
1	0	0	0	1	0
1	0	0	1	0	0
1	0	0	1	1	0
1	0	1	0	0	0
1	0	1	0	1	0
1	0	1	1	0	0
1	0	1	1	1	0
1	1	0	0	0	0
1	1	0	0	1	0
1	1	0	1	0	0
1	1	0	1	1	0
1	1	1	0	0	0
1	1	1	0	1	0
1	1	1	1	0	0
1	1	1	1	1	0
1	1	1	1	1	1

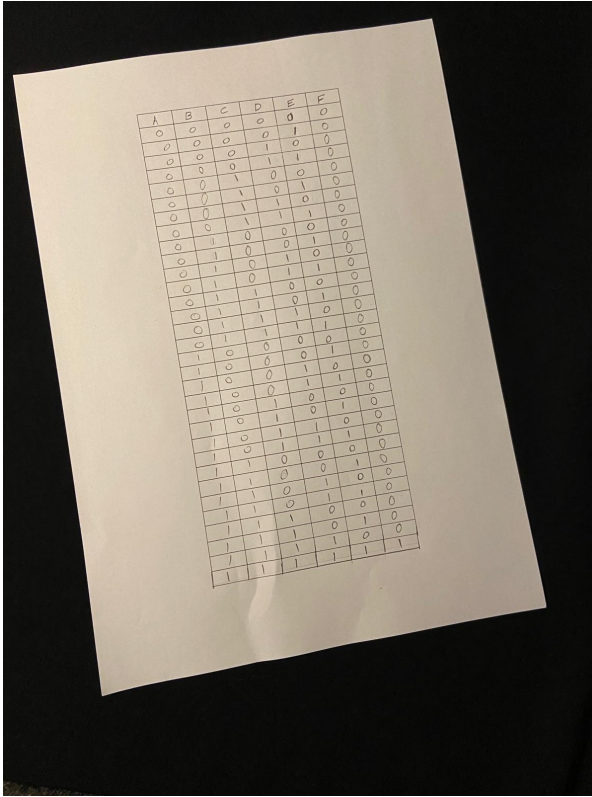
rows numbers 33

cols numbers 6

num_outputs: 1

F0 = abcde +

Failed test cases:



The classifier detected 2 of the digits wrong So it added more terms in the generated sum of products

```
rows numbers 33
cols numbers 6
num_outputs: 1
F0 = a'bc'd'e' + abc'd'e' + abcde
```

References:

- <https://docs.opencv.org/4.x/>
- Burger, W. (n.d.). Principles of Digital Image. Hagenberg.
- Chlamtac, I. (n.d.). Text Segmentation and Recognition for Enhanced Image Spam Detection. Tiruchirappalli, India: Bharathidasan University.
- Gonzalez, R. C. (2007). Processing, Digital Image. Pearson.
- Kristin P Bennett, C. C. (n.d.). Support Vector Machines.