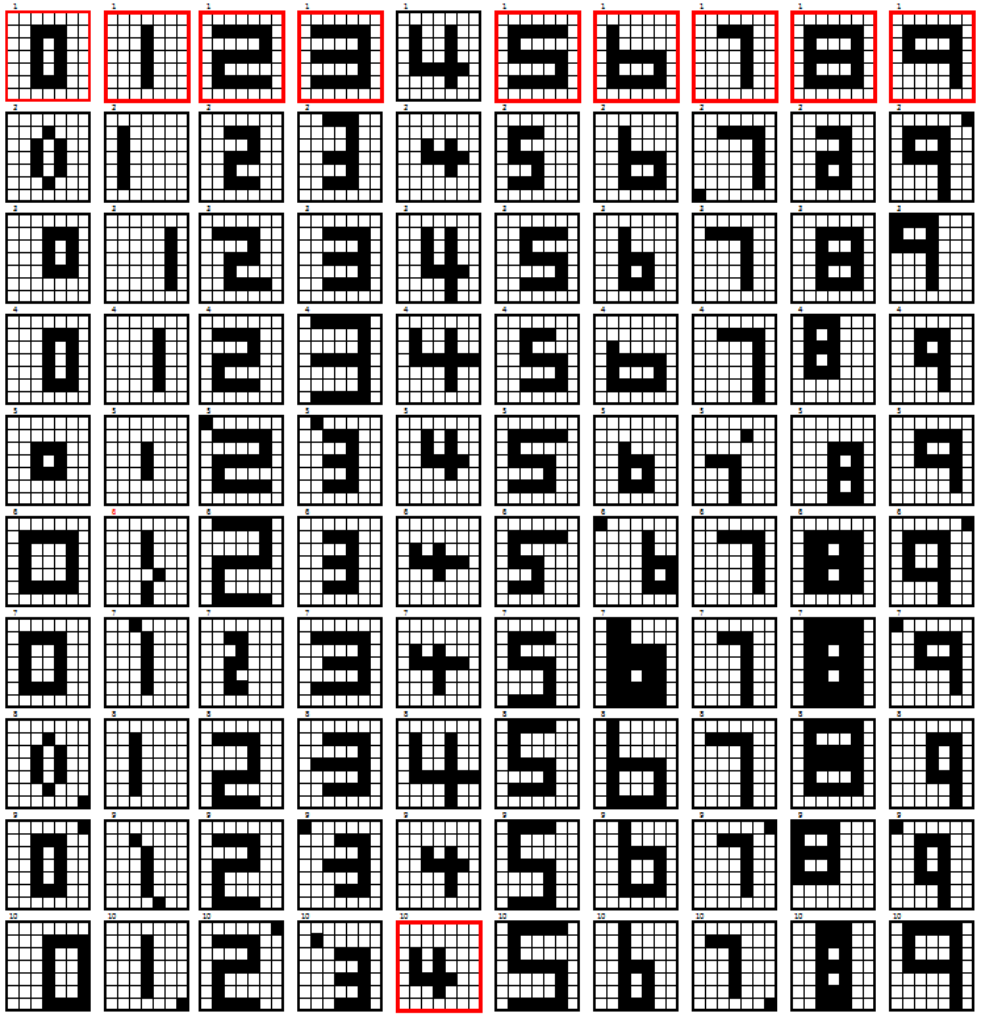
인공지능

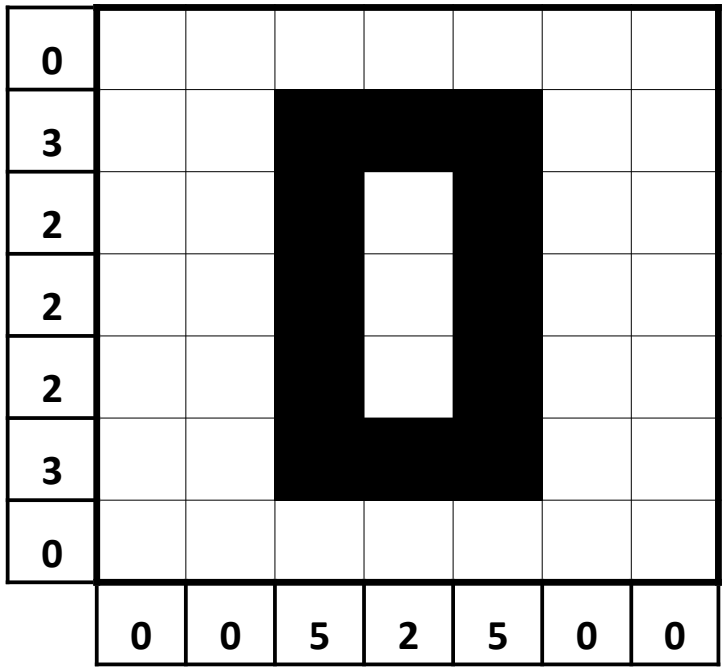
- 숫자인식 프로젝트 -

지도 교수님: 공용해 교수님
제출일: 2019.05.21.
소속: 순천향대학교 의료IT공학과
성명: 20165215 위예진

1. 숫자샘플 정하기

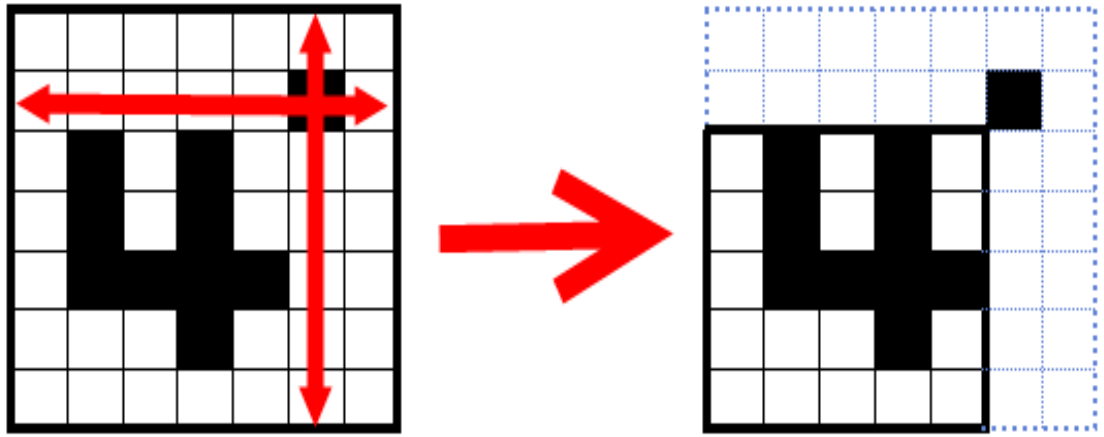


2. 기본이 되는 특징 구하기




```
#include "help.h"
void counting(int N[SORT][ORDER][ROWS][COLS], int sort, int order)
{

        // 개수 초기화
        for (int i = 0; i < ROWS; i++) {
                count_r[i] = 0;
                count_c[i] = 0;
        }
        // 행마다 1의 개수를 셈
        for (int i = 0; i < ROWS; i++) {
                for (int j = 0; j < COLS; j++) {
                        if (N[sort][order][i][j] == 1)
                                count_r[i] += 1;
                        else
                                ;
                }
        }
        // 열마다 1의 개수를 셈
        for (int j = 0; j < COLS; j++) {
                for (int i = 0; i < ROWS; i++) {
                        if (N[sort][order][i][j] == 1)
                                count_c[j] += 1;
                        else
                                ;
                }
        }
}
```

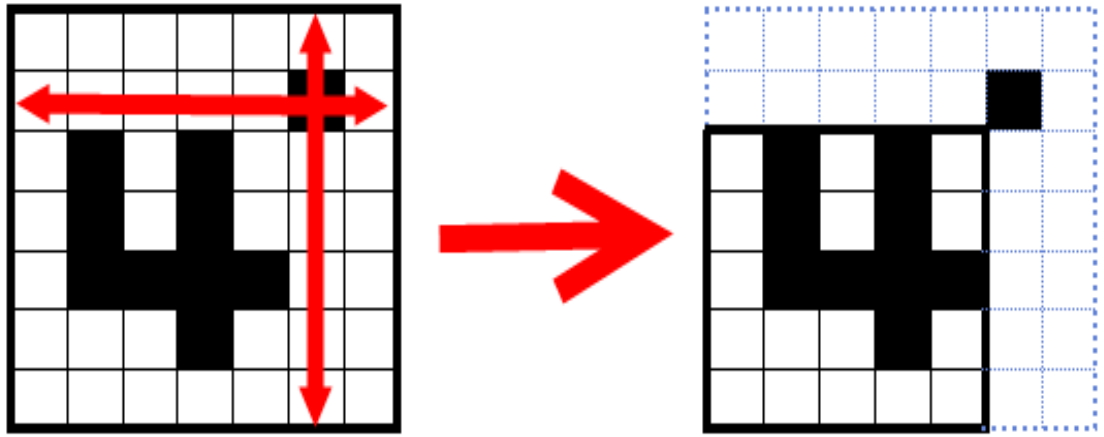
3. 잡음 제거



```
// 가로, 세로의 각각의 합이 1인 행, 열도 제외(십자 중에 합이 1이면 1)
int cross_r(int N[SORT][ORDER][ROWS][COLS], int sort, int order, int row) {
		int col;
		if (count_r[row] == 1) {
			//1이 있는 열의 위치 찾음
			for (int m = 0; m < COLS; m++) {
				if (N[sort][order][row][m] == 1) {
					col = m;
					break;
				}
				else ;
			}
			//열의 1개수가 1인지
			if (count_c[col] == 1)
				return 1;
			else
				return 0;
		}
		else
			return 0;
}
```

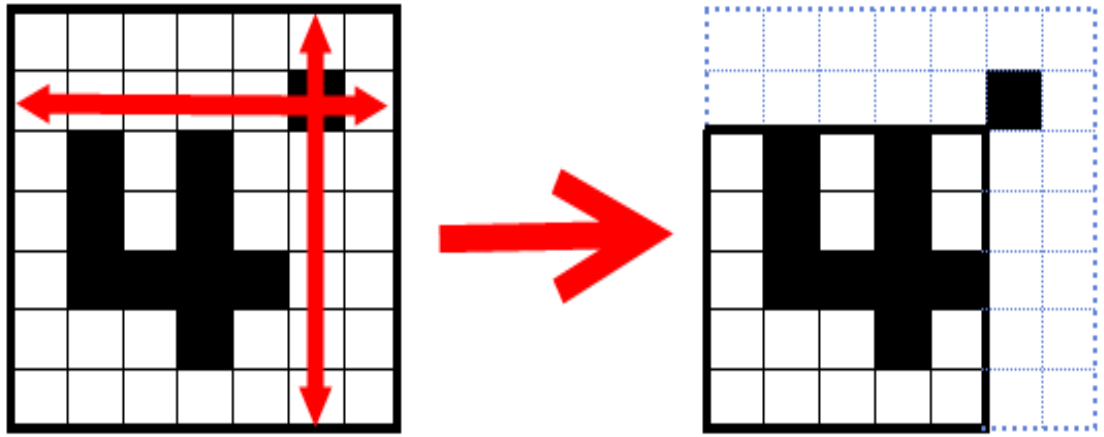
```
int cross_c(int N[SORT][ORDER][ROWS][COLS], int sort, int order, int
col) {
		int row;
		if (count_c[col] == 1) {
			//1이 있는 행의 위치 찾음
			for (int m = 0; m < ROWS; m++) {
				if (N[sort][order][m][col] == 1)


{
					row = m;
					break;
				}
				else;
			}
			//행의 1개수가 1인지
			if (count_r[row] == 1)
				return 1;
			else
				return 0;
		}
		else

			return 0;
}
```

3. 잡음 제거



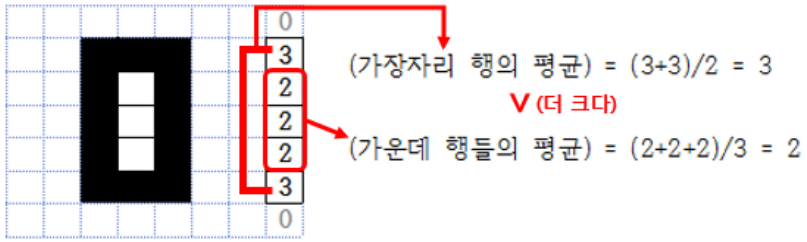
```
void downsize(int N[SORT][ORDER][ROWS][COLS], int sort, int order) {
        // 1의 개수합이 0인 행 인식계산에서 제외하기
        for (int i = 0; i < ROWS; i++) {
                if (count_r[i] != 0 && cross_r(N, sort, order, i) == 0) {
                        row_s = i;
                        for (int j = i; j < ROWS; j++) {
                                if (cross_r(N, sort, order, j) == 1) {
                                        row_e = j - 1;
                                        break;
                                }
                                else if (count_r[j] == 0 && cross_r(N, sort, order, j)
== 0) {
                                        row_e = j - 1;
                                        break;
                                }
                                else if (count_r[6] != 0 && cross_r(N, sort, order,
6)==0) {
                                        row_e = 6;
                                        break;
                                }
                        }
                        break;
                }
        }
```

3. 잡음 제거



```
// 1의 개수합이 0인 열 인식계산에서 제외하기
for (int i = 0; i < COLS; i++) {
                if (count_c[i] != 0 && cross_c(N, sort, order, i) == 0) {
                                col_s = i;
                                for (int j = i; j < COLS; j++) {
                                                if (cross_c(N, sort, order, j) == 1) {
                                                                col_e = j - 1;
                                                                break;
                                                }
                                                else if (count_c[j] == 0) {
                                                                col_e = j - 1;
                                                                break;
                                                }
                                                else if (count_c[6] != 0 && cross_c(N, sort, order, 6) == 0)
{
                                                                col_e = 6;
                                                                break;
                                                }
                                }
                                break;
                }
        }
}
```

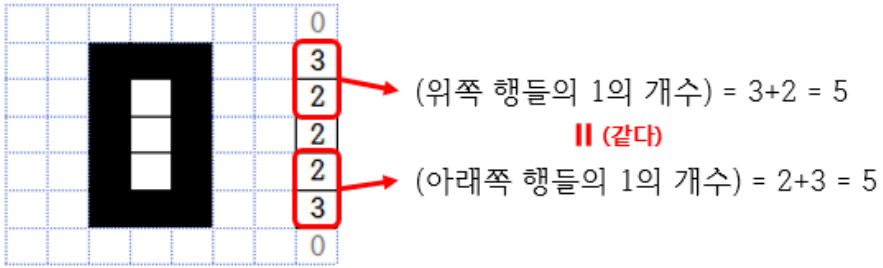
4. 특징벡터 정하기 – (1)



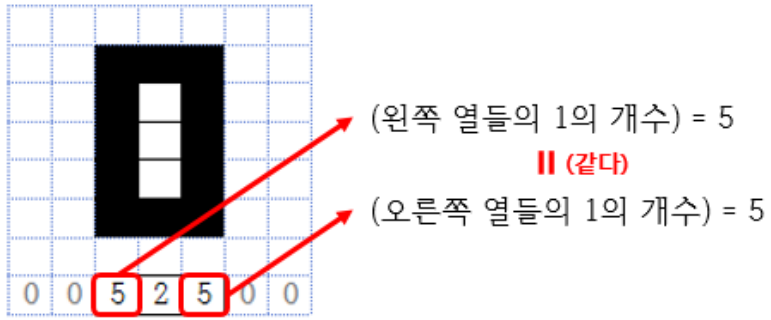
∴ (특징 벡터 값) = 1

```
//맨위아래행과 가운데행들의 평균 1의 개수비교(위아래행 평균이 더 크면 1)
int avgCom(int R[ROWS], int start, int end) {
        double count_edge = (R[start] + R[end]) / 2.0;
        double count_center = 0.0;
        int center_num = end - start - 1;
        for (int i = 0; i < center_num; i++) {
                count_center += R[start + 1 + i];
        }
        count_center /= center_num;
        if (count_edge > count_center)
                return 1;
        else if (count_edge < count_center)
                return -1;
        else
                return 0;
}
```

4. 특징벡터 정하기 – (2, 3)



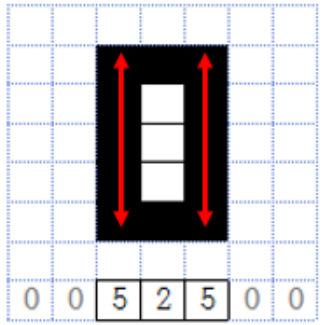
∴ (특징 벡터 값) = 0



∴ (특징 벡터 값) = 0

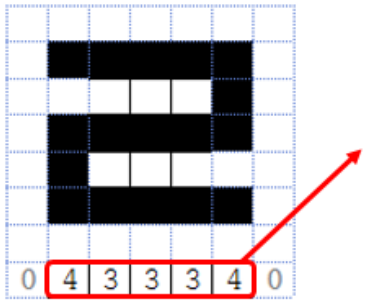
```
//행,열을 반으로 나눠 1의 개수 비교(위쪽이 더 큰가?)
int half(int R[ROWS], int start, int end) {
        int count_l = 0;
        int count_r = 0;
        for (int i = 0; i < MID(start, end); i++) {
                count_l += R[start + i];
                count_r += R[end - i];
        }
        if (count_l == count_r)
                return 0;
        else if (count_l > count_r)
                return 1;
        else
                return -1;
}
```

4. 특징벡터 정하기 – (4)



꽉 차있는 열이 존재함 (2개 있음)

∴ (특징 벡터 값) = 0

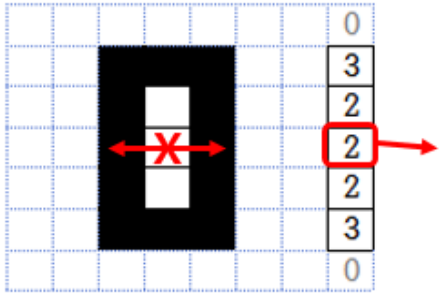


꽉 차있는 열이 존재하지 않음
(5가 없음)

∴ (특징 벡터 값) = 0

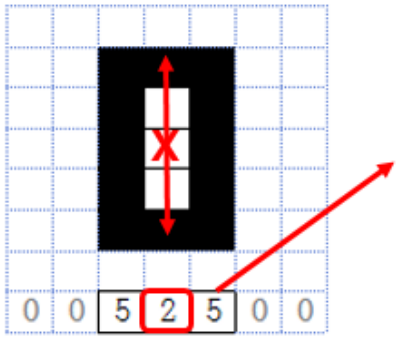
```
/100%로 차있는 열이 없음(없으면 1)
int noFullCol(int C[COLS], int start, int end, int r_s, int
    ) {
        int count = 0;
        for (int i = 0; i < end - start + 1; i++) {
            if (C[start + i] == r_e - r_s + 1)
                count += 1;
            else
                ;
        }
        if (count > 0)
            return 0;
        else
            return 1;
}
```

4. 특징벡터 정하기 – (5, 6)



행의 중간이 꽉 차지 않음
(3이 아님)

∴ (특징 벡터 값) = 0

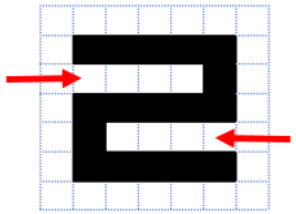


열의 중간이 꽉 차지 않음
(5가 아님)

∴ (특징 벡터 값) = 0

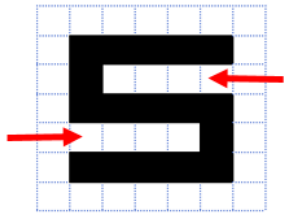
```
//행, 열의 중간이 100% 차있음(차있으면 1)
int midFull(int D[ROWS], int start, int end, int s,
int e) {
        int mid = MID(start, end) + start;
        if (D[mid] == e - s + 1)
                return 1;
        else
                return 0;
}
```

4. 특징벡터 정하기 – (7)



첫번째 행이 먼저!

∴ (특징 벡터 값) = 1



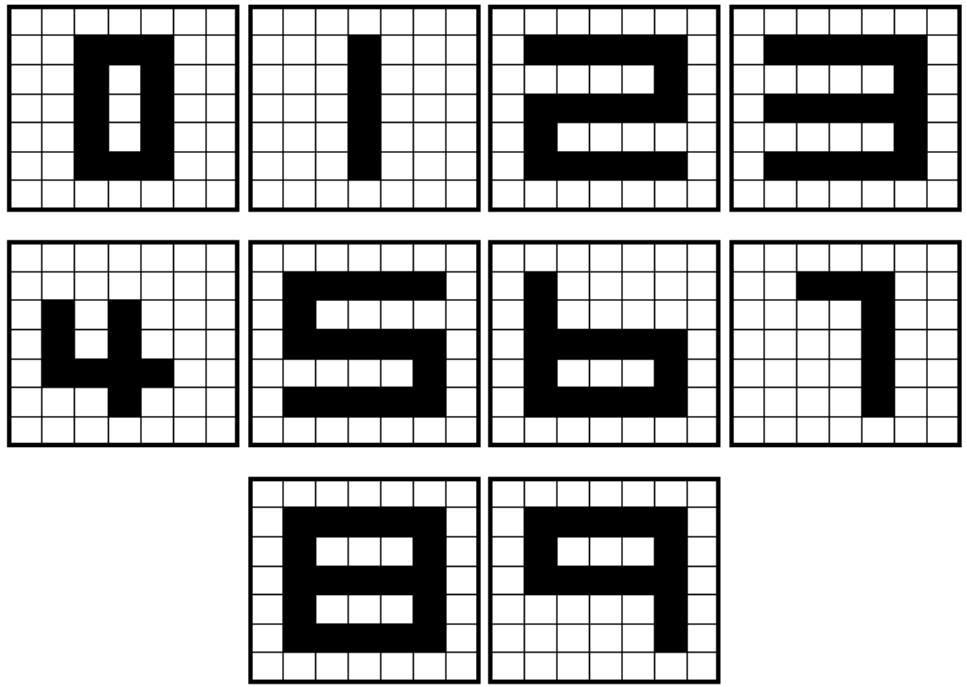
마지막 행이 먼저!

∴ (특징 벡터 값) = -1

```
//숫자 2,5 구분을 위한 앞, 뒤 뚫려있는 거 찾기(둘의 행위치 달라야 함)
int first0(int D[SORT][ORDER][ROWS][COLS], int r, int c, int row_s, int row_e, int
col_s, int col_e) {
		int left = 10;
		int right = 10;

		//왼쪽 위부터
		for (int i = 0; i < row_e - row_s-1; i++) {
			if (D[r][c][row_s + i + 1][col_s] == 0) {
				left = row_s + i + 1;
				break;
			}
		}
		//오른쪽 아래부터
		for (int i = 0; i < row_e - row_s - 1; i++) {
			if (D[r][c][row_e - i - 1][col_e] == 0) {
				right = row_e - i - 1;
				break;
			}
		}
		if (left == 10 || right == 10)
			return 0;
		else if (left < right && left != 10 && right != 10)
			return 1;
		else if (right < left && left != 10 && right != 10)
			return -1;
		else
			return 0;
}
```

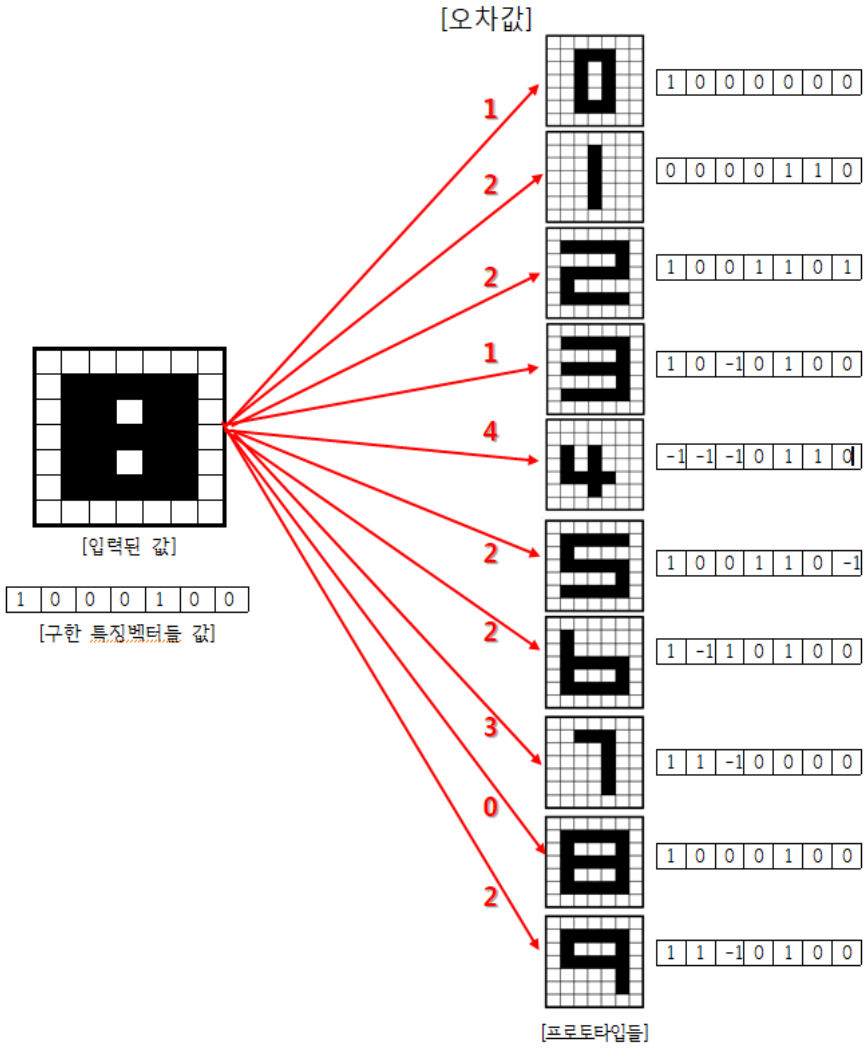
5. 프로토타입 정하기

특징벡터 함수의 이름	0	1	2	3	4	5	6	7	8	9
avgCom (위아래행평균이더크면1, 작으면-1)	> (1)	=	> (1)	> (1)	< (-1)	> (1)	▽ (>)	> (1)	> (1)	▽ (>)
half (R) (위쪽이더크면1,작으면-1)	=	=	=	=	▽ (<)	=	< (-1)	> (1)	=	> (1)
half (C) (왼쪽이더크면1,작으면-1)	=	=	=	<	▽ (<)	=	> (1)	< (-1)	=	< (-1)
noFullCol (없으면1)	O (0)	O (0)	X (1)	O (0)	O (0)	X (1)	O (0)	O (0)	O (0)	O (0)
midFull (R) (차있으면1)	X (0)	O (1)	O (1)	O (1)	▽ (0)	O (1)	▽ (0)	X (0)	O (1)	▽ (0)
midFull (C) (차있으면1)	X (0)	O (1)	X (0)	X (0)	▽ (0)	X (0)	X (0)	X (0)	X (0)	X (0)
first0(2,5) (왼쪽에서찾은게더먼저임2 그러면1)	X (0)	X (0)	O (1)	X (0)	X (0)	O (-1)	X (0)	X (0)	X (0)	X (0)



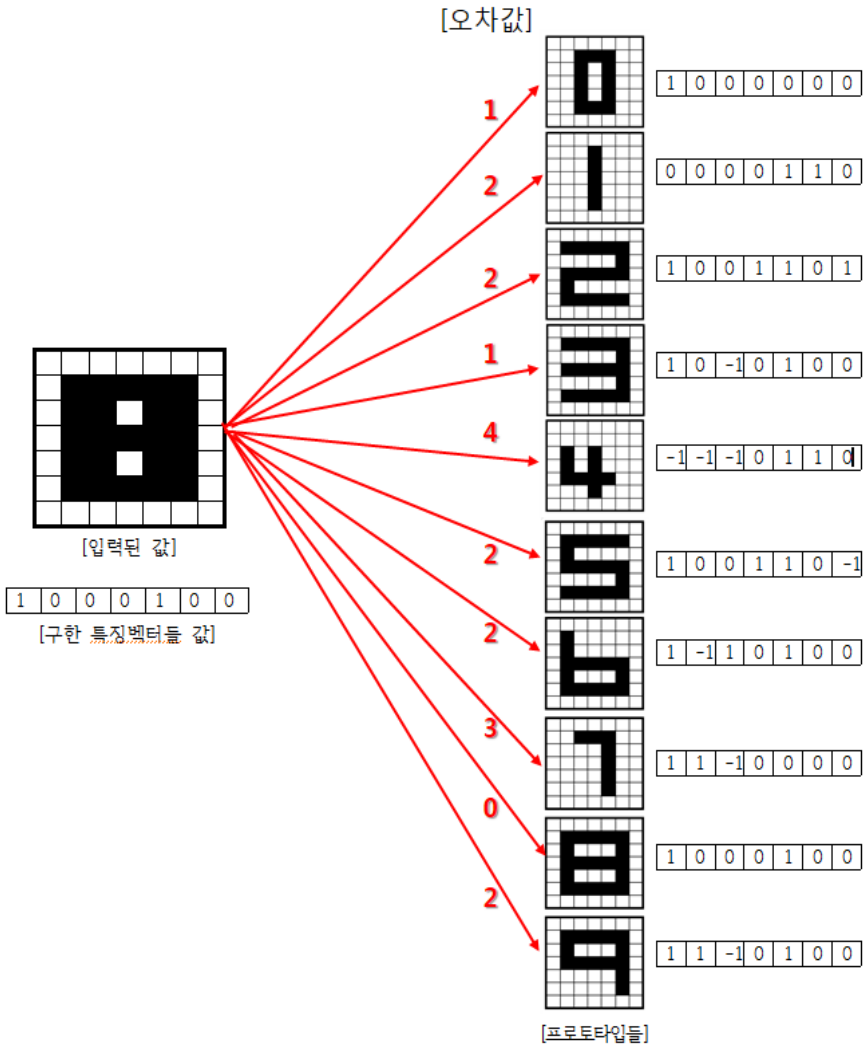
[그림. 숫자 0~9의 각각의 프로토타입]

6. 인식 방법 (프로토타입과 비교)




```
//* 숫자 인식하는 곳 -> main
#include "help.h"
#include <stdio.h>
void main() {
		//각 숫자들의 프로토타입의 특징벡터 추출
		//열의 값
		//avgCom, half(R), half(C), noFullCol, midFull(R),midFull(C) 순서
		/*
		int avgCom(int R[ROWS], int row_s, int row_e);
		int half(int R[ROWS], int start, int end);
		int noFullCol(int C[COLS], int start, int end, int r_s, int r_e);
		int midFull(int D[ROWS], int start, int end, int s, int e);
		*/
		printf("--프로토타입의 특징벡터 배열--₩n");
		for (int i = 0; i < SORT; i++) {
			counting(protoData, i, 0);
			downsize(protoData, i, 0);

			featureNum[i][0] = avgCom(count_r, row_s, row_e);
			featureNum[i][1] = half(count_r, row_s, row_e);
			featureNum[i][2] = half(count_c, col_s, col_e);
			featureNum[i][3] = noFullCol(count_c, col_s, col_e, row_s,
row_e);
			featureNum[i][4] = midFull(count_r, row_s, row_e, col_s, col_e);
			featureNum[i][5] = midFull(count_c, col_s, col_e, row_s, row_e);
			featureNum[i][6] = first0(protoData, i, 0, row_s, row_e, col_s,
col_e);
			printf("숫자 %d : (%d, %d, %d, %d, %d, %d, %d) ₩n", i,
featureNum[i][0], featureNum[i][1], featureNum[i][2], featureNum[i][3], featureNum[i][4],
featureNum[i][5], featureNum[i][6]);
			rate[i] = 0;
		}
		printf("₩n");
		//전체 일치율 초기화
		totalRate = 0;
```

6. 인식 방법 (프로토타



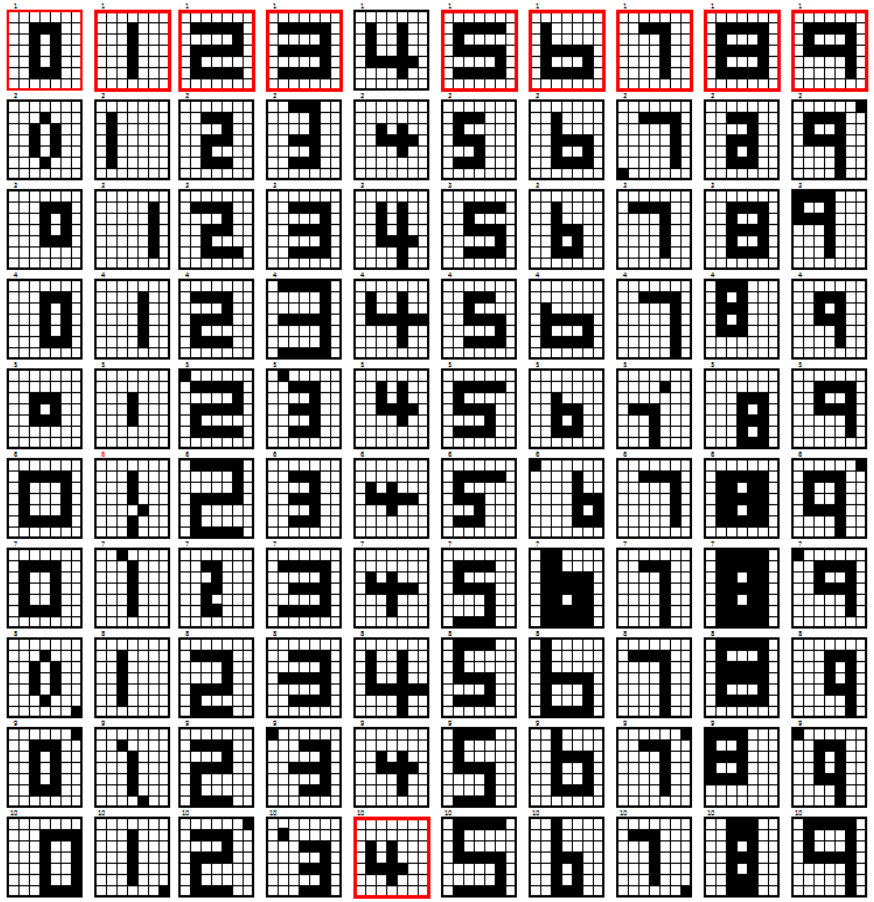

```
//100개의 숫자들 입력
for (int i = 0; i < SORT; i++) {
                for (int j = 0; j < ORDER; j++) {
                                counting(inData, i, j);
                                downsize(inData, i, j);

                                featureIn[0] = avgCom(count_r, row_s, row_e);
                                featureIn[1] = half(count_r, row_s, row_e);
                                featureIn[2] = half(count_c, col_s, col_e);
                                featureIn[3] = noFullCol(count_c, col_s, col_e, row_s, row_e);
                                featureIn[4] = midFull(count_r, row_s, row_e, col_s, col_e);
                                featureIn[5] = midFull(count_c, col_s, col_e, row_s, row_e);
                                featureIn[6] = first0(inData, i, j, row_s, row_e, col_s, col_e);
                                for (int m = 0; m < SORT; m++) {
                                                for (int n = 0; n < FEATURE; n++) {
                                                                if       (featureNum[m][n]       ==
featureIn[n])
                                                                                rate[m] += 1;
                                                }
                                }
                                for (int m = 0; m < SORT; m++) {
                                                if (rate[m] > rate[max])
                                                                max = m;
                                }
                                if (max == i)
                                                numRate += 1;
                                else
                                                ;
                                printf("%d", max);
                                //프로토타입과의 일치율, 일치숫자 초기화
                                for (int m = 0; m < SORT; m++)
                                                rate[m] = 0;
                                max = 0;
                                //printf("₩n");
                                printf(" ");
                }
                totalRate += numRate;
                printf("숫자 %d의 일치율 : %d₩n", i, numRate*10);
                //숫자일치율 초기화
                numRate = 0;
}
printf("전체 숫자인식률 : %d ₩n₩n₩n₩n₩n", totalRate);

}
```

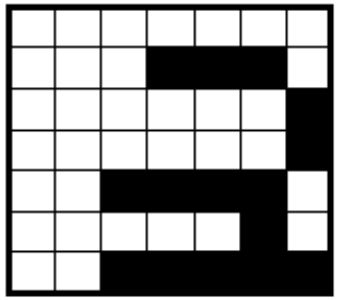
7. 인식률

구분	0	1	2	3	4	5	6	7	8	9	총합
인식률	92.8	88.1	98.1	52.2	58.4	66.3	80.6	94.7	54.3	57.5	74.3

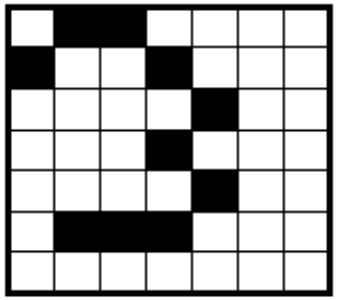
학생	0	1	2	3	4	5	6	7	8	9	총합
1	60	70	100	10	0	60	90	100	40	20	55
2	100	100	100	90	80	80	100	100	70	40	86
3	100	100	100	90	30	90	100	100	60	70	84
4	100	100	100	90	100	70	100	100	90	90	94
5	80	70	100	30	60	30	60	80	10	30	55
6	80	70	100	30	60	30	60	80	10	30	55
7	100	100	100	100	90	90	90	100	50	70	89
8	90	80	90	20	70	10	40	90	20	20	53
9	100	90	90	0	40	30	50	40	0	0	44
10	100	100	100	100	50	100	100	100	80	90	92
11	100	80	100	40	80	70	80	100	60	60	77
12	100	90	100	50	100	80	60	100	60	20	76
13	100	100	100	70	60	60	90	100	50	90	82
14	90	90	100	90	50	100	100	100	100	70	89
15	100	100	100	60	60	80	70	100	90	80	84
16	90	100	100	10	0	60	90	100	60	90	70
17	90	90	90	40	80	70	80	100	70	80	79
18	50	50	100	10	40	60	40	100	20	30	50
19	100	100	100	50	80	90	80	100	50	60	81
20	100	90	100	80	90	100	90	100	60	50	86
21	100	100	100	90	90	50	80	100	50	60	82
22	100	60	90	40	50	60	80	100	50	10	64
23	50	60	100	0	100	10	50	50	0	30	45
24	100	100	100	70	10	70	100	100	100	90	84
25	100	100	100	10	0	60	90	100	60	90	71
26	100	80	90	50	50	60	70	90	40	30	66
27	90	60	90	50	30	90	100	100	60	60	73
28	100	90	100	100	100	60	80	100	60	90	88
29	100	100	100	50	80	70	90	100	70	60	82
30	100	100	100	10	0	60	90	100	60	90	71
31	100	100	100	50	80	70	90	100	70	60	82
32	100	100	100	90	60	100	90	100	70	80	89

8. 고찰 (인식률이 낮은 이유)

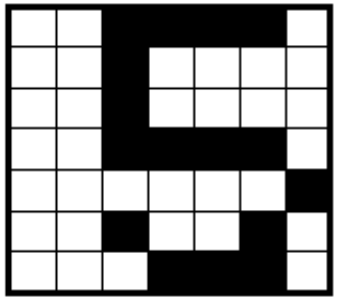




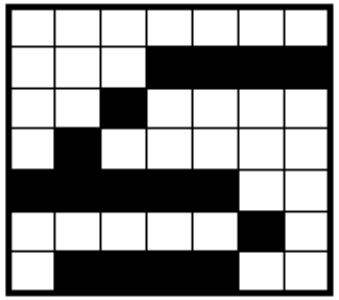
[학생 1의 숫자 3의 5번째]



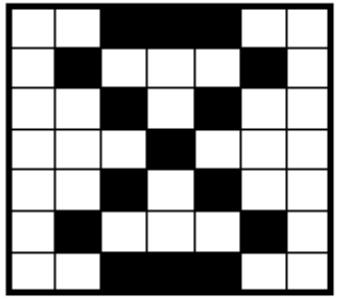
[학생 9의 숫자 3의 9번째]



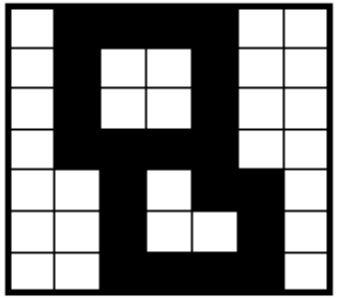
[학생 23의 숫자 5의 10번째]



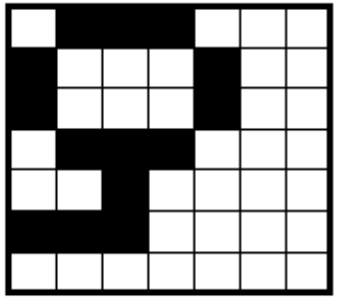
[학생 8의 숫자 5의 4번째]



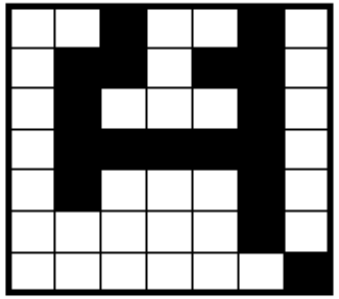
[학생 18의 숫자 8의 8번째]



[학생 18의 숫자 8의 6번째]



[학생 8의 숫자 9의 9번째]



[학생 1의 숫자 9의 9번째]