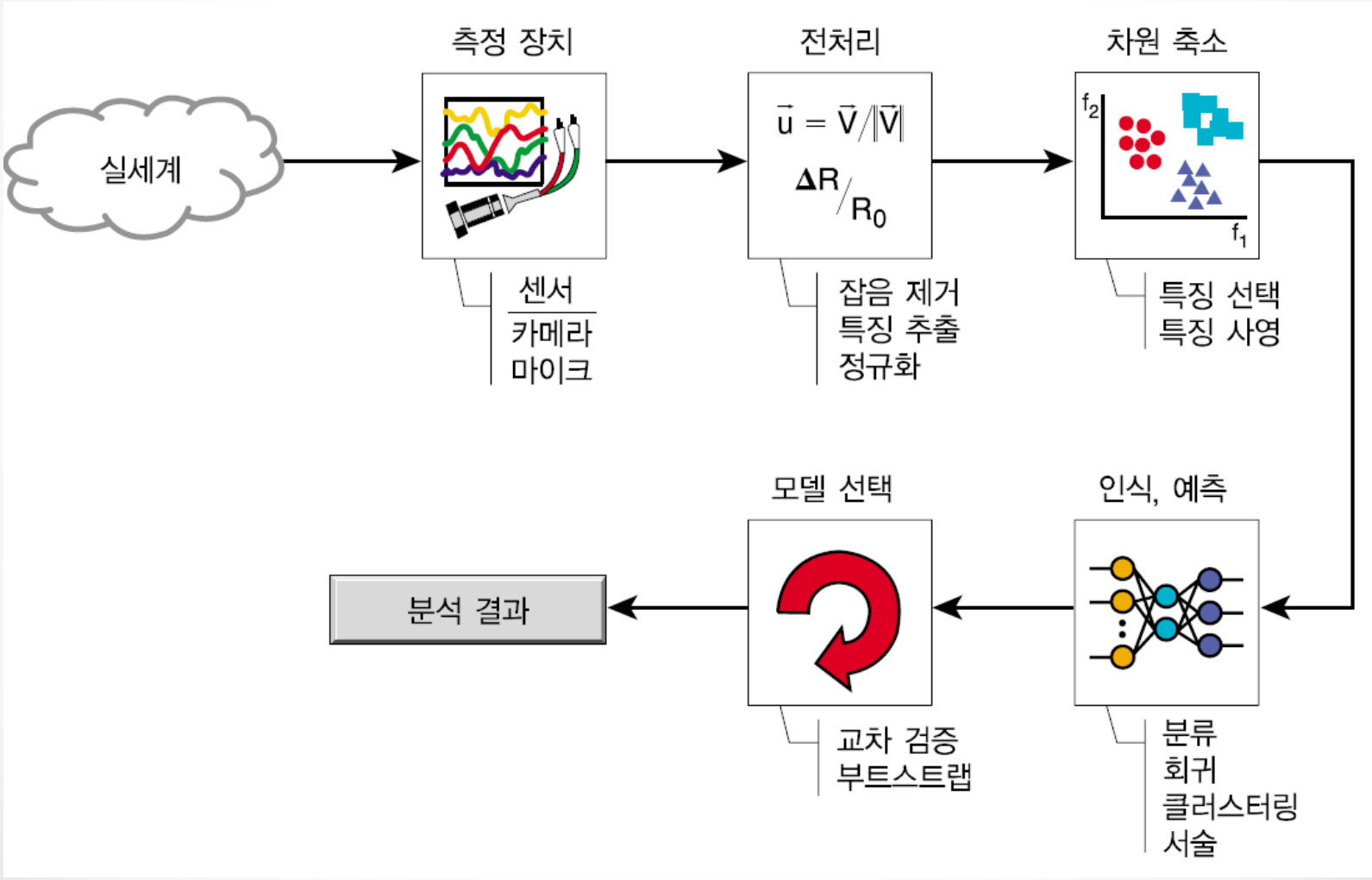
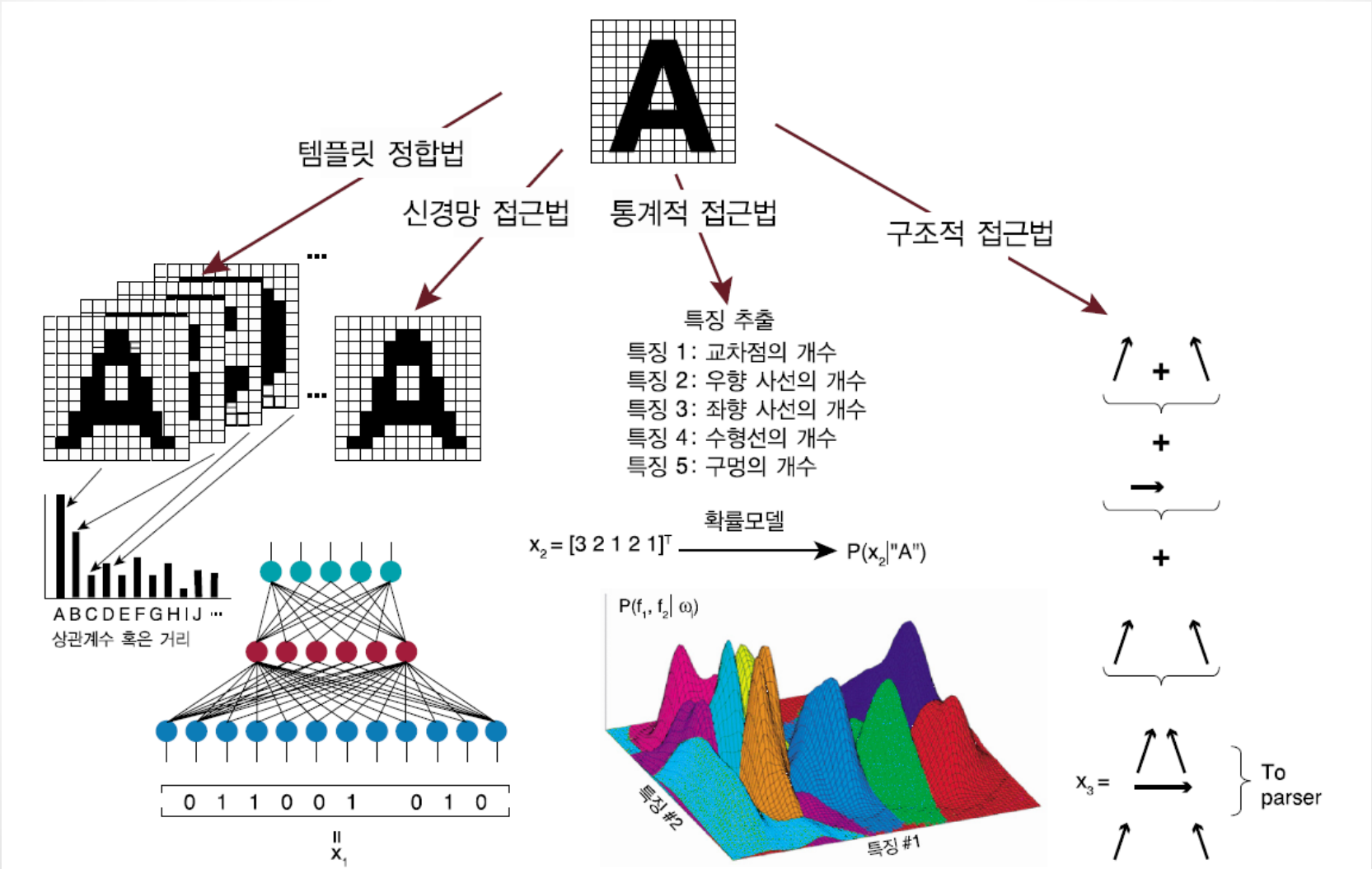
Python

Application 1

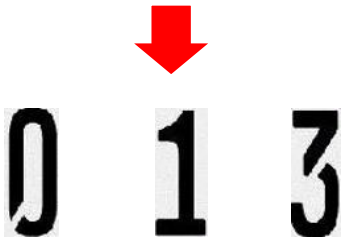
Application



Application



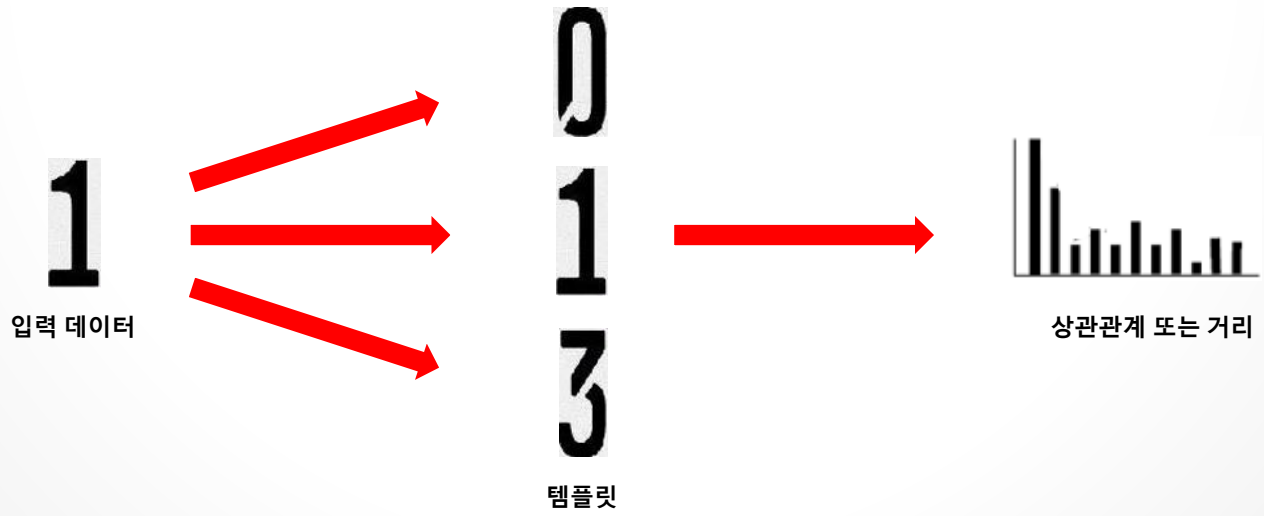
Application



분석 결과

Application

- 템플릿 정합법
  - 패턴인식에서 가장 오래되고 쉬운 접근법.
  - 비교 대상 패턴에 대한 템플릿을 미리 마련해두고, 인식하고자 하는 패턴을 템플릿 구성 조건에 맞추는 정규화 과정을 거쳐서 상호상관 혹은 거리와 같은 유사도를 척도로 하여 패턴을 인식하는 방법.



Application

- Exercise 1
  - 숫자 0~5 데이터집합을 이용해 템플릿을 생성하여 분류기를 만들어보자!!
    1. 각 숫자의 평균 이미지를 생성
    2. 생성된 평균 이미지 집합과 입력 숫자 이미지를 각각 비교
    3. 차이가 가장 적은 템플릿을 선택하여 출력

Application

```
import numpy as np
import matplotlib.image as img

path = "C:\\Users\\hault\\바탕 화면\\testData\\"
trainData = np.zeros((20 * 16, 7))
meanData = np.zeros((20 * 16, 5))


for i in range(0, 5, 1):
    for j in range(1, 7, 1):
        fn = path + "digit%d_%d.bmp" % (i, j)
        xi = img.imread(fn)
        x = xi[:, :, 0].reshape(16 * 20) / 255
        trainData[:, j] = x
    meanData[:, i] = np.mean(trainData, 1)
```

Application

```
k = input("파일명을 입력하세요 : ")
fn = path + k
x = img.imread(fn)
data = x[:, :, 0].reshape(16*20) / 255

distArray = []

for i in range(0, 5, 1):
    dist = np.linalg.norm(meanData[:, i] - data)
    distArray.append(dist)

minVal = distArray.index(min(distArray))

print("입력한 데이터는 %d 입니다" % (minVal))
```