AI Programming

Lab 01: Numpy for arrays

소속: 컴퓨터정보공학부

학번: 2019202103

이름: 이은비

제출날짜: 23/09/16

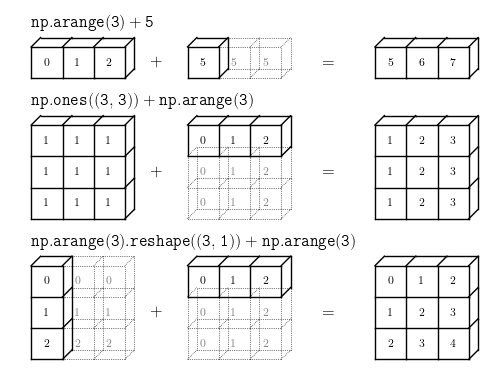
텍스트, 로고, 폰트, 등록 상표이(가) 표시된 사진

자동 생성된 설명

**<Lab Objective>**

첨부된LAB\_01\_Numpy\_for\_Arrays\_dist자료와 수업자료를 참고하여 numpy패키지에서 array에 대해 여러가지 operations과 functions을 적용합니다. 주어진 seed를 기반으로 한 random array를 작성합니다. Numpy의 내장된 random함수를 적용하여 코드를 작성합니다. 그리고 broadcast를 이용한 matrix의 multiplication (“ \* ”)을 진행합니다. 이때 일반적인 matrix multiplication (“ @ ”)과 비교해서 다르게 진행되는 이유를 알고, 그에 따른 결과를 reshape하는 과정을 진행합니다. 이후 argmax와 max를 통해서 np의 functions을 진행하도록 합니다. 또한 위의 과정을 통해 얻어낸 max\_index를 통과시킴으로써 one hot인코딩 작업을 진행할 수 있도록 합니다. 이번 과제를 통해서는 numpy의 array, 즉, matrix를 만들고 shape을 다르게 하여 변화를 주는 것, 최대값을 직접 뽑아내는 것, 최대값의 index를 뽑아내는 것, 그리고 특정조건을 만족하는 one-hot encoding의 작업을 개념적 이해와 그에 맞는 코드를 작성할 수 있도록 합니다.

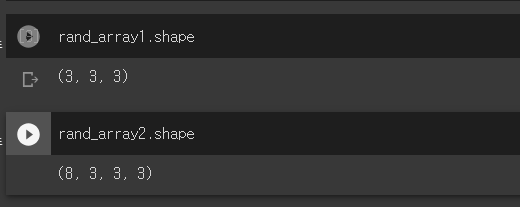
**<Whole simulation program flow>**

Numpy package를 import합니다. 또한 이후에 np.random.rand(,,)의 인자로 들어오는 option값에 따라서 그에 맞는 차원,depth와 row, col을 기존에 제시된 값으로 initialize하는 과정을 거칩니다. 또한 이후 np.random.rand(,,)즉 numpy의 random함수를 통해서 option값에 맞는 각 array를 만들도록 하고, 생성된 array를 “ \* ” 라는 operator를 통해 자동 broadcast후 matrix의 multiplication을 진행 할 수 있도록 합니다. 이때 사용되는 broadcast에 대해서 추가적으로 알아보면 다음 그림과 함께 설명될 수 있는데 

위의 그림에서 첫 번째 그림에서는 “np.arange(3)”로 0이상 3미만에 해당하는0, 1, 2라는 Numpy로 생성한 배열에 스칼라 5를 합한 결과가 5, 6, 7이라는 것을 알 수 있습니다. Numpy가 아닌 phython에서의 리스트를 사용하면 for문을 이용해야 위와 같은 결과를 얻을 수 있지만, Numpy에서는 앞서 언급한 broadcast 개념 덕분에 5가 0이외에 1과 2의 원소 부분에도 계산되어 간단하게 합 연산을 수행하는 것만으로 같은 결과를 얻을 수 있었습니다. 두 번째 그림은 배열 간의 계산으로 배열의 차원이 확대된 케이스입니다. 그리고 이 그림은 3x3 배열에 1x3 배열을 합 연산한 경우입니다. 세번째 그림은3x1 배열과 1x3 배열의 합의 결과가 짝에 맞게 계산되기 위해 두개의 배열 모두 broadcast한 것을 확인할 수 있었습니다.

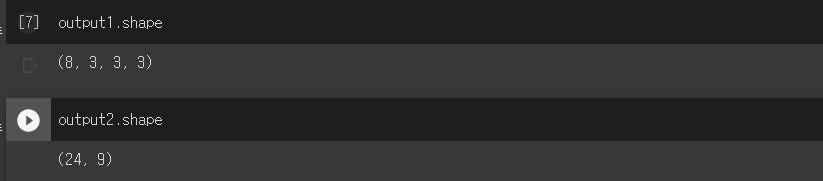
위의 내용을 정리하면 Broadcasting은 모양이 다른 배열들 간의 연산이 어떤 조건을 만족했을 때 가능해지도록 배열을 자동적으로 변환하는 것이라고 정의할 수 있습니다. 그리고 누락되었거나 길이가 1인 차원에 대해서도 Broadcasting 이 수행되는 것을 알 수 있었습니다. 그리고 이러한 Broadcasting 은 numpy에서 자동으로 수행되므로 사용자가 원하는 상황인지를 항상 검토해야 되며 Broadcasting이 일어날 수 있는 조건은 다음과 같이 정리할 수 있습니다. 첫째, 차원의 크기가 1일때 가능합니다.  
두 배열 간의 연산에서 최소한 하나의 배열의 차원이 1일때 가능하며,

두번째, 차원의 짝이 맞을 때 가능합니다.  
차원에 대해 축의 길이가 동일하면 Broadcasting 이 가능합니다. 위의 조건을 만족할 때 Broadcasting이 가능합니다. 그리고 수행한 과제 예제에서는 두번째 조건을 만족한 경우이며



위의 그림과 같이 진행된 것을 확인할 수 있었습니다.

그리고 그 결과로 만들어진 output1이라는 random array를 reshape한 결과로 출력되는 output2는 아래 캡쳐 된 그림과 같이 reshape과정을 거치기 전(output1.shape)과 후(output2.shape)를 .shape을 통해 확인할 수 있었습니다.



그리고 이처럼 reshape과정을 거친 output2에 대해서 argmax와 max를 통해서 그 array에 대한 구조와 index, 값을 확인할 수 있도록 합니다. 그리고 max index의 값을 기반으로 통과시켜서 그 값을 인자로 받는 one\_hot인코딩을 진행하고 각 array에 대한 각 과정에 대한 결과를 print함수를 통해 확인합니다.

**<Your codes & comments on your codes (English only)>**

import numpy as np # import the package "numpy"

# set the option values

depth = 3

np\_row = 3

np\_column = 3

output\_num = 8

np.random.seed(622) # seed value is 622

### START CODE HERE ###

# random seed [0,1)

rand\_array1 = np.random.rand(depth,np\_row,np\_column) # get rand array [0,1) (3, 3, 3)

rand\_array2 = np.random.rand(output\_num,depth,np\_row,np\_column)

# get rand array [0,1) (8, 3, 3, 3)

output1 = rand\_array1 \* rand\_array2 # calculate rand\_array1 \* rand\_array2 with broadcast

output2 = output1.reshape(-1, np\_row\*np\_column) # reshape the output1 shape ( ,np\_row\*np\_column)

max\_index = np.argmax(output2, axis=-1) # get max\_index (axis = -1)

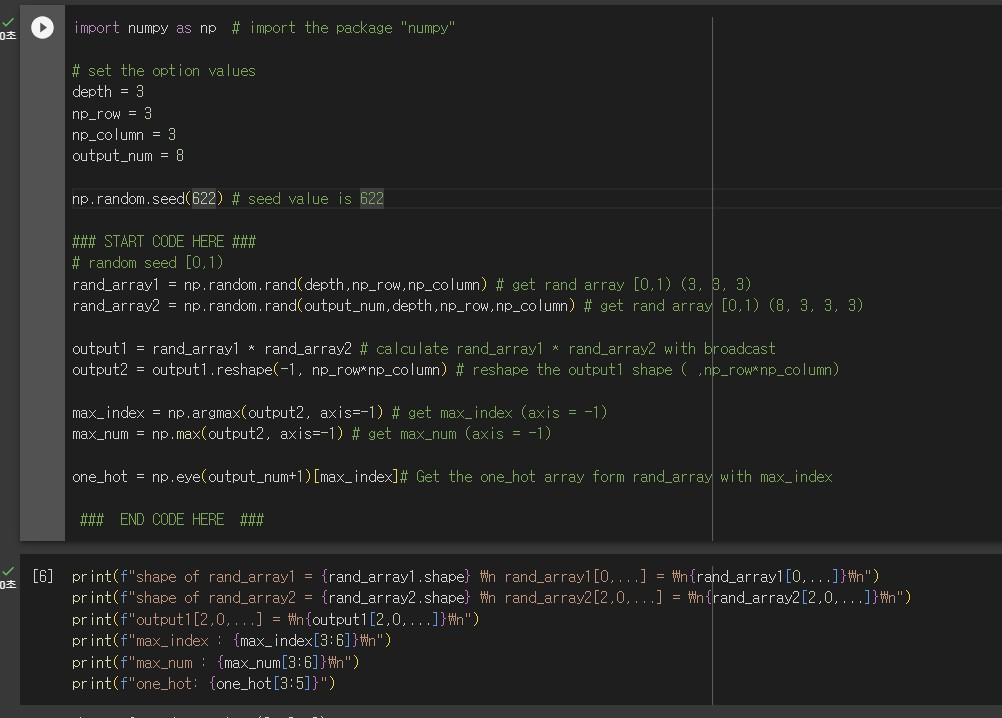
max\_num = np.max(output2, axis=-1) # get max\_num (axis = -1)

one\_hot = np.eye(output\_num+1)[max\_index]

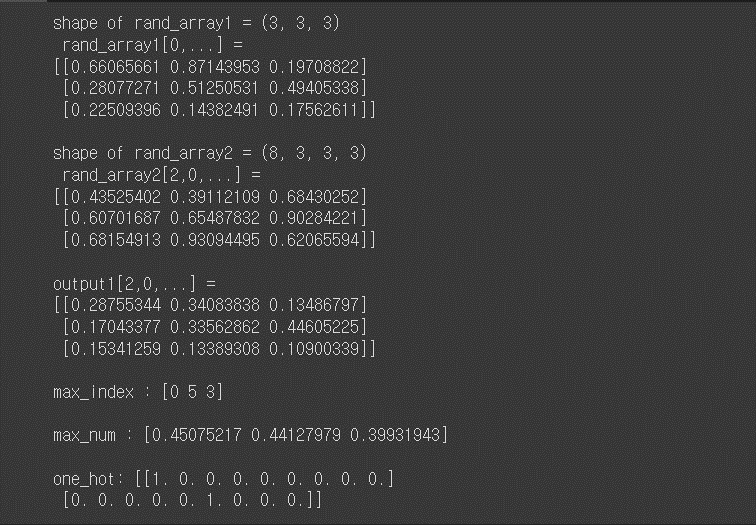
# Get the one\_hot array form rand\_array with max\_index

###  END CODE HERE  ###

**<Simulation results>(screenshots)**



전체 코드내용을 구글 코랩에서 진행한 것입니다.



random함수를 통한 random array1, random array2의 형태와 그 값을 확인하는 위의 2개의 결과 출력과 그 다음 output1의 결과 array와 max\_index와 max\_num의 function들을 지난 후 확인된 결과, 그리고 위에서 언급한 것처럼 one\_hot 인코딩의 과정을 겪은 후 one\_hot 인코딩의 결과까지 확인할 수 있었습니다.

**<Discussion>**

처음 random함수를 생성할 때 정해진 범위조건과 분포의 종류에 따라서 np.rand와 np.randn을 다르게 적용할 수 있는 것을 수업자료에서 확인하였습니다. 따라서 과제에서 사용한 np.random.rand()는 0부터 1이전까지의 균일 분포에서의 난수 matrix array를 생성하는 function이고, 그 외에 np,random.randn()은 가우시안 표준 정규 분포에서 난수 matrix array를 생성하는 function이며 np.random.randint()는 균일 분포의 정수 난수 1개를 생성하는 function입니다. 그리고 이때 random함수를 생성하기 이전에 random seed라고 seed값을 지정해서 그에 맞게 값을 산출하는데 이때 seed값이 고정되어 있다면 random이라고 해도 나오는 패턴이 고정된 형태라고 하여 그 원인을 확인하니 랜덤 seed값이 랜덤하게 나오는 것이 아니라 시간을 기준으로 출력되는 값이기 때문이었습니다. 이것이 마치 랜덤한 숫자처럼 보이는 이유는 ms초로 결정이 되다 보니 seed 값들이 랜덤하게 보이는 것이었습니다. 그리고 마지막으로 one\_hot인코딩을 진행할 때 출력되는 값으로 사용되는 단위행렬에 대해서 eye를 사용하였는데 eye함수는 숫자를 인자로 받고 해당 숫자만큼의 행을 갖는 단위행렬을 생성합니다. 따라서 과제 관련 수정사항이 없을 때는 one\_hot = np.eye(output\_num)[max\_index]로 작성하였으나 이후 출력되는 개수가 달라져서 one\_hot = np.eye(output\_num+1)[max\_index]로 수정하는 과정을 거쳤습니다.