머신러닝/딥러닝을 위한

TensorFlow 기초 (I)

– 텐서 • 상수 • 변수 • 플레이스홀더 –

TensorFlow – 설치

- TensorFlow는 Google에서 개발하고 공개한 머신러닝/딥러닝 라이브러리
  - C++, Java 등의 다양한 언어를 지원하지만 파이썬(Python)에 최적화 되어있음

- 윈도우 환경에 TensorFlow 설치
  - 아나콘다 배포판* 으로 파이썬 설치하고, 파이썬 pip 를 이용하여 TensorFlow 설치 가능

```
(C:₩Program Files₩Anaconda3) C:₩Users₩SungHoPark>pip install tensorflow
Requirement already satisfied: tensorflow in c:₩program files₩anaconda3₩lib₩site-packages (1.12.0)
```

※ 아나콘다 배포판 다운로드 : https://www.anaconda.com/distribution/

※ TensorFlow 설치 시 error 가 나는 경우에는, ① python -m pip install --upgrade pip 명령으로 pip 버전을 업그레이드 한 후 ② pip install --ignore-installed --upgrade tensorflow 실행

※ 아나콘다 5.3 버전부터 파이썬 3.7 버전을 사용하도록 변경됨. 그러나 2019년 현재 TensorFlow는 파이썬 3.7 을 지원하지 않기 때문에 아나콘다 5.3 버전을 설치하면 파이썬 3.6 으로 다운그레이드해야 함 ⇒ 그러므로 파이썬 3.6을 사용하는 아나콘다 5.2 버전을 아래의 사이트에서 직접 다운받아 설치하는 것이 필요함

https://repo.anaconda.com/archive/Anaconda3-5.2.0-Windows-x86_64.exe

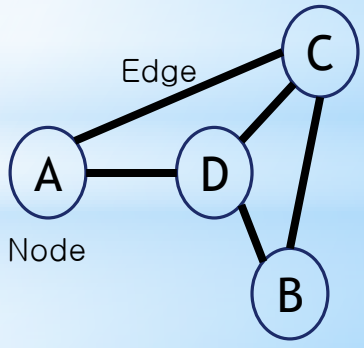
TensorFlow – 텐서 / 그래프 (노드, 엣지)

- TensorFlow는 이름이 나타내고 있는 것처럼 텐서(Tensor)를 흘려보내면서(Flow) 머신러닝과 딥러닝 알고리즘을 수행하는 라이브러리임.
  - 숫자 1 (스칼라 또는 rank 0 텐서)
  - 1차원 배열 [1, 2] (벡터 또는 rank 1 텐서)
  - 2차원 배열 [[1,2], [3,4]] (행렬 또는 rank 2 텐서)
  - 3차원 배열 [[[1,2]], [[3,4]]] (텐서 또는 rank 3 텐서)

모든 데이터는 텐서로 취급함

상수, 변수, 텐서연산(+,-, 행렬곱, 컨벌루션연산등) 나타냄

- 이러한 텐서들은 그래프(Graph) 구조에서 노드(Node)에서 노드로 흘러감(Flow)
  - 그래프 자료구조는 노드(Node)와 엣지(Edge)로 구성됨
  - 텐서플로를 이용한 프로그램 작성 시,
    ① 상수, 변수, 텐서연산 등의 노드와 엣지를 먼저 정의하고,
    ② 세션을 만들고 그 세션을 통해 노드간의 데이터(텐서) 연산 수행



그래프구조
(4개 노드, 5개 엣지)

TensorFlow 상수 노드 – tf.constant(…)

상수 값을 저장하는 노드를 만들기 위해서 상수 노드 tf.constant 정의

①

세션을 만들지 않고 print와 같은 명령문을 실행하면, 저장된 값이 아닌 현재 정의되어 있는 노드의 상태 (노드타입, shape 등) 출력됨

②

③

노드간의 연산을 위해 세션 생성

④

세션을 통해 (sess.run()) 노드에 값이 할당되고 노드간의 텐서를 흘려보내면서(tensor flow) 연산과 명령문 등이 실행됨

⑤

생성된 세션 close

시각화 툴인 텐서보드에서 name 지정

```
import tensorflow as tf

# 상수 노드 정의
a = tf.constant(1.0, name='a')
b = tf.constant(2.0, name='b')
c = tf.constant([ [1.0, 2.0], [3.0, 4.0] ])

print(a)
print(a+b)
print(c)


# 세션 (session) 을 만들고 노드간의 텐서 연산 실행
sess = tf.Session()

print(sess.run([a, b]))
print(sess.run(c))
print(sess.run([a+b]))
print(sess.run(c+1.0))  # broadcast 수행

# 세션 close
sess.close()
```

```
Tensor("a:0", shape=(), dtype=float32)
Tensor("add:0", shape=(), dtype=float32)
Tensor("Const:0", shape=(2, 2), dtype=float32)
[1.0, 2.0]
[[1. 2.]
 [3. 4.]]
[3.0]
[[2. 3.]
 [4. 5.]]
```

TensorFlow 플레이스홀더 노드 – tf.placeholder(…)

텐서플로에서는 임의의 값을 입력으로 받기 위해 플레이스홀더 노드(tf.placeholder) 정의

①

노드간의 연산을 위해 세션 생성

②

플레이스홀더 노드에 실제 값을 넣어줄때는 sess.run 첫번째 인자로 실행하고자 하는 연산을 넣어주고, 두번째 인자에는 실제로 넣을 값들을 Dictionary 형태로 넣어주는 feed_dict을 선언하고, feed_dict 부분에 플레이스홀더에 넣을 값을 지정해줌

③

생성된 세션 close

④

```python
import tensorflow as tf

# 플레이스홀더 노드 정의
a = tf.placeholder(tf.float32)
b = tf.placeholder(tf.float32)
c = a + b


# 세션 (session) 을 만들고 플레이스홀더 노드를 통해 값 입력받음
sess = tf.Session()

print(sess.run(c, feed_dict={a: 1.0, b: 3.0}))
print(sess.run(c, feed_dict={a: [1.0, 2.0], b: [3.0, 4.0]}))

# 연산 추가
d = 100 * c

print(sess.run(d, feed_dict={a: 1.0, b: 3.0}))
print(sess.run(d, feed_dict={a: [1.0, 2.0], b: [3.0, 4.0]}))


# 세션 close
sess.close()
```

실행하고자 하는 연산

플레이스홀더 노드에 실제 대입되는 값

```
4.0
[4. 6.]
400.0
[400. 600.]
```

플레이스홀더 노드는 머신러닝/딥러닝에서
입력데이터 (input), 정답데이터 (target) 를 넣어주기 위한 용도로 주로 사용됨

TensorFlow 변수 노드 – tf.Variable(…)

가중치나 바이어스처럼 계속 업데이트 되는 변수는 텐서플로에서 변수 노드 (tf.Variable) 로 정의*

①

노드간의 연산을 위해 세션 생성

②

변수노드 값 초기화를 위해서 반드시 tf.global_variables_initializer() 실행함

③

변수노드 값 업데이트

④

생성된 세션 close

⑤

tf.Variable(…) 에서 사용되는 초기값*

tf.random_normal, tf.truncated_normal,

tf.random_uniform, tf.ones, tf.zeros,

tf.constant 등이 있음

```python
import tensorflow as tf

# 값이 계속 업데이트되는 변수노드 정의
W1 = tf.Variable(tf.random_normal([1]))  # W1 = np.random.rand(1) 비슷함
b1 = tf.Variable(tf.random_normal([1]))  # b1 = np.random.rand(1) 비슷함

W2 = tf.Variable(tf.random_normal([1,2]))  # W2 = np.random.rand(1,2) 비슷함
b2 = tf.Variable(tf.random_normal([1,2]))  # b2 = np.random.rand(1,2) 비슷함

# 세션 생성
sess = tf.Session()

# 변수노드 값 초기화. 변수노드를 정의했다면 반드시 필요함
sess.run(tf.global_variables_initializer())


for step in range(3):

    W1 = W1 - step    # W1 변수노드 업데이트
    b1 = b1 - step    # b1 변수노드 업데이트

    W2 = W2 - step    # W2 변수노드 업데이트
    b2 = b2 - step    # b2 변수노드 업데이트

    print("step = ", step, ", W1 = ", sess.run(W1), ", b1 = ", sess.run(b1))
    print("step = ", step, ", W2 = ", sess.run(W2), ", b2 = ", sess.run(b2))

# 세션 close
sess.close()
```

```
step =  0 , W1 =  [-0.45137885] , b1 =  [1.2115546]
step =  0 , W2 =  [[ 0.5276252  -0.19593444]] , b2 =  [[ 1.0040597  -0.63668317]]
step =  1 , W1 =  [-1.4513788] , b1 =  [0.21155465]
step =  1 , W2 =  [[-0.4723748 -1.1959344]] , b2 =  [[ 0.00405967 -1.6366832 ]]
step =  2 , W1 =  [-3.4513788] , b1 =  [-1.7884454]
step =  2 , W2 =  [[-2.472375  -3.1959343]] , b2 =  [[-1.9959403 -3.6366832]]
```