Deep Learning practice #3-2 report

컴퓨터소프트웨어학부 2018008768 윤정

[구현 과정]

```
W1 = np.array([[random.uniform(0, 1)],[random.uniform(0, 1)]]) #(2, 1)
B1 = np.array([[random.uniform(0, 1)], [random.uniform(0, 1)]])
W2 = np.array([[random.uniform(0, 1), random.uniform(0, 1)]]) #(1, 2)
B2 = np.array([[random.uniform(0, 1)]])
```

주어진 뉴럴 네트워크에 맞게 학습 parameter를 설정한 다음, 2-layer back propagation을 통해 파라미터를 update시키며 학습을 진행하였습니다.

이 과정에서, 계산상의 에러 방지 및 더 좋은 성능을 위해 정규화를 진행하였습니다. degree value인 x값을 radian value값으로 변환한 뒤 train, predict를 진행하였습니다.

구한 parameter W1, B1, W2, B2 값에 대하여 Cost와 Accuracy를 계산하였습니다.

[결과]

m = 10000, n = 1000, K = 5000, alpha = 0.9 일 때 결과값 (매 500회마다 출력, 최종 w와 b)

(중간 결과는 생략하였습니다)

```
0 th result----
layer 1 parameter :
w1_1:  0.34821473121183794 b1_1:  0.8846196224972612 w1_2:  0.2621412675668102 b1_2:  0.8912429570705447
layer 2 parameter :
w2_1:  0.1707408952820486 w2_2:  0.2815033219006576 b2:  0.3199482461003393
500 th result----
layer 1 parameter :
w1_1:  3.089967022225667 b1_1:  -4.58443122138369 w1_2:  1.1452546352694823 b1_2:  -3.2216486617277966
layer 2 parameter :
w2_1:  -5.811636181244471 w2_2:  4.433959569369206 b2:  2.434162267047821
1000 th result----
layer 1 parameter :
w1_1:  4.384105158435037 b1_1:  -6.9727111042819905 w1_2:  2.6803045407023 75 b1_2:  -10.653427071603197
layer 2 parameter :
w2_1:  -9.238838399076366 w2_2:  7.559211654193189 b2:  4.403419004224708
1500 th result----
layer 1 parameter :
w1_1:  5.071238409652017 b1_1:  -8.01621325982289 w1_2:  3.318588885292419 b1_2:  -13.971368177991632
layer 2 parameter :
w2_1:  -10.679628332416534 w2_2:  8.904943722354217 b2:  5.205392485537755
```

```
4000 th result----
layer 1 parameter :
w1_1:  6.833576247707238 b1_1:  -10.70955504226992 w1_2:  4.711266994641704 b1_2:  -21.144268121523925
layer 2 parameter :
w2_1:  -13.702269945840621 w2_2:  11.774837922322492 b2:  6.940217139615674
4500 th result----
layer 1 parameter :
w1_1:  7.065110362214533 b1_1:  -11.069228385123303 w1_2:  4.876964893668672 b1_2:  -21.979129363129935
layer 2 parameter :
w2_1:  -14.06778610563476 w2_2:  12.12929216596746 b2:  7.139613994496459
--------------------
layer 1 parameter :
w1_1:  7.2771353615756675 b1_1:  -11.398501842813754 w1_2:  4.7676364868010666 b1_2:  -22.779178571744065
layer 2 parameter :
w2_1:  -14.417926883298739 w2_2:  12.441090433585016 b2:  7.298396268816115
Train result
Cost:  0.04986182113617408
Accuracy:  98.06
Test result
Cost:  0.061520433502893324
Accuracy:  96.89999999999999
```

alpha값을 변화시키며 예측한 결과값입니다. (M = 10000, N = 1000, k = 5000)

	alpha=0.01	alpha=0.05	alpha=0.1	alpha=0.5	alpha=0.9
Cost (with 'n' test samples)	0.66571248 84744788	0.56482016 59352344	0.31672636 326796605	0.50097061 33169815	0.0498346 9016204
Accuracy (with 'n' test samples)	73.8	71.5	96.3	72.8999	97.8

가장 작은 cost를 가지는 0.9를 본 실험에서 알파 값으로 사용하였습니다.

Train data m을 변화시키며 예측한 결과값입니다. (alpha=0.9)

	m=10, n=1000, K=5000	m=100, n=1000, K=5000	m=10000, n=1000, K=5000
Accuracy (with 'm' train samples)	100	96.0	98.06
Accuracy (with 'n' test samples)	72.1	94.6	96.8999999999

K를 변화시키며 예측한 결과값입니다. (alpha = 0.9)

	m=10000, n=1000, K=10	m=10000, n=1000, K=100	m=10000, n=1000, K=5000
Accuracy (with 'm' train samples)	50.26	72.84	98.06
Accuracy (with 'n' test samples)	48.9	73.0	96.899999999

[분석]

작은 네트워크에서는 W의 초기 값이 생각보다 결과에 영향을 미칠 수 있다는 것을 알게 되었습니다. 처음 실험에서는 모든 파라미터의 기본값을 1로 주고 시작했는데, 생각보다 학습이 원활하게 이루어지지 않았고, 정확도도 많이 떨어졌습니다. 그래서 각 w의 초기 값을 random.uniform(0, 1)을 통해 0과 1 사이의 랜덤 값으로 주니, 학습의 정확도가 훨씬 상승한 것을 볼 수 있었습니다.

W와 b값은 학습을 통해 계속 update되기 때문에 큰 영향을 주지 않을 것이라고 생각했는데, 이번 실험을 통해 이 또한 학습에 큰 영향을 미칠 수 있다는 것을 알게 되었습니다.

Hidden unit이 한 개였던 저번 실험에서는 정확도가 70퍼센트대가 나왔는데, hidden unit을 두 개로 만들면서 정확도가 거의 100에 가깝게 올라갔습니다. 이는 cos 그래프의 개형 때문에 hidden unit이 두 개일 때가 더 학습에 적합하기 때문이라고 생각했습니다.