부산대학교 전기컴퓨터공학부 정보컴퓨터공학전공  
201724579 정현모

1. **개요**

하이퍼 파라미터 및 모델 파라미터를 튜닝하여 성능 개선을 위해 작업을 하였다. 기본적으 로 전복의 고리 수 추정을 위한 신경망 모델의 실험을 진행하였다. 하이퍼 파라미터는 모델 구 조나 학습 과정에 영향을 미치는 각종 상숫값 (학습률, 학습횟수, 미니배치 크기)을 의미하며, 개발자가 미리 정해주어야하고, 최적의 값을 찾을 수 있어야 좋은 결과를 기대할 수 있다.

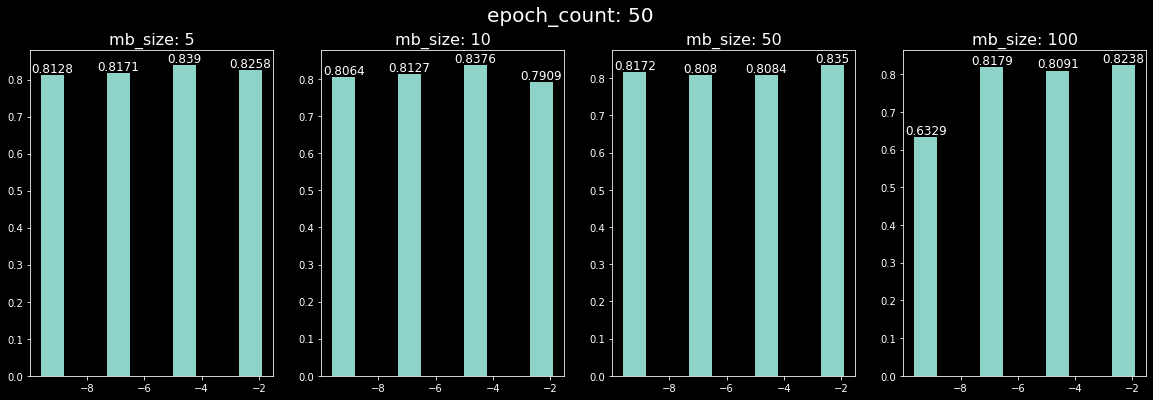
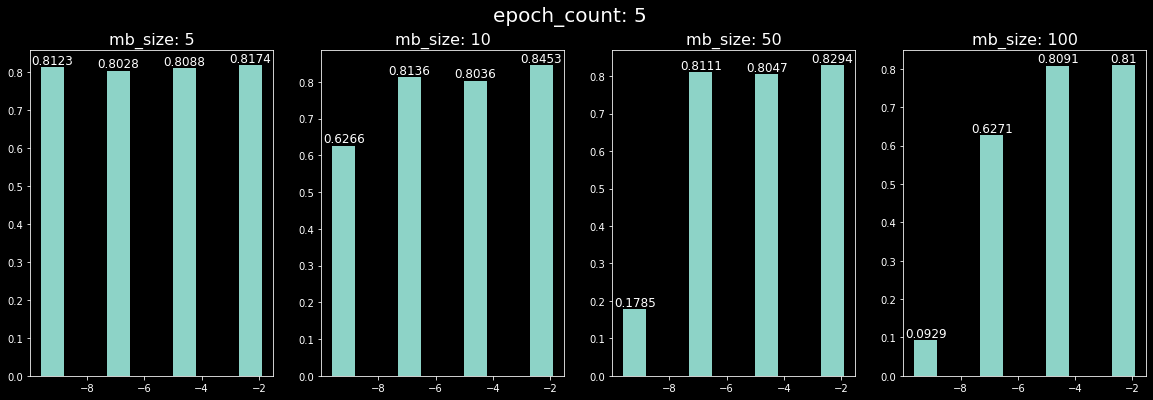
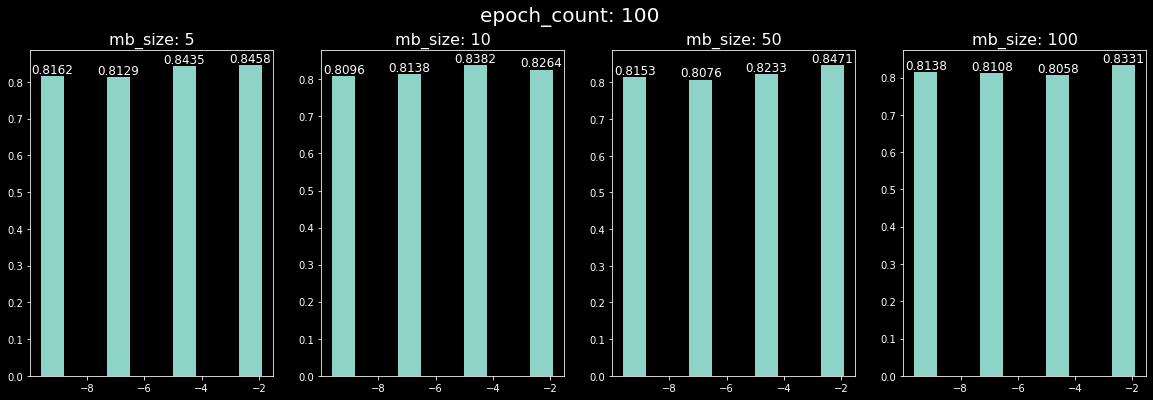
1. **코드 변경**

학습 과정에 영향을 미치는 하이퍼 파라미터를 epoch\_count, mb\_size, learning\_rate 이렇게 3개로 잡았으며, 하이퍼 파라미터 변화에 따른 결과값을 한눈에 보기 위해 코드를 수정하였다.  
 코드 수정 내용은, abalone\_exec() 함수에 인자로 learning\_rate를 추가하였고, 학습결과를 한눈에 보기 위해 그래프로 도출하였다. 그래프로 학습결과를 확인하기 위해 최종 정확도를 return하도록 abalone\_exec() 함수와 train\_and\_test() 함수를 수정하였다.

1. **파라미터별 학습 결과**

아래 X축은 learning\_rate로 [0.0001, 0.001, 0.01, 0.1]을 사용하였고, 그래프에서 보기 편하도록 np.log()를 사용해 간격을 맞춰주었다. Epoch가 높을 수록 대체로 학습 정확도가 좋았으며, 미니배치 사이즈는 10에서 대체로 정확도가 좋았으며, learning\_rate는 어느정도 클수록 학습 정확도가 좋았다.   
 학습 정확도는 아래와 같을 때 가장 최대가 되었다.

|  |  |
| --- | --- |
| epoch\_count | 5 |
| mb\_size | 10 |
| learning\_rate | 0.1 |
| final\_accuracy | 0.8561468646 |

1. **수정한 코드**

import matplotlib.pyplot as plt

epoch\_counts = [5, 10, 50, 100]

mb\_sizes = [5, 10, 50, 100]

learning\_rates = [0.0001, 0.001, 0.01, 0.1]

maximum = {"epoch\_count": -1,

 "mb\_size": -1,

 "learning\_rate": -1,

 "final\_accuracy": 0}

for epoch\_count in epoch\_counts:

  fig = plt.figure(figsize=(20,6))

  fig.suptitle("epoch\_count: " + str(epoch\_count), fontsize=20)

  for i, mb\_size in enumerate(mb\_sizes):

    plt.subplot(1, len(mb\_sizes), i + 1)

    plt.title("mb\_size: " + str(mb\_size), fontsize=16)

    final\_accuracy = []

    for j, learning\_rate in enumerate(learning\_rates):

      final\_accuracy.append(

        abalone\_exec(epoch\_count = epoch\_count,

                     mb\_size = mb\_size,

                     learning\_rate = learning\_rate))

      if final\_accuracy[j] > maximum["final\_accuracy"]:

        maximum["final\_accuracy"]  = final\_accuracy[j]

        maximum["epoch\_count"] = epoch\_count

        maximum["mb\_size"] = mb\_size

        maximum["learning\_rate"] = learning\_rate

    # print(learning\_rates, final\_accuracy)

    plt.bar(np.log(learning\_rates), final\_accuracy)

    for i, v in enumerate(np.log(learning\_rates)):

      plt.text(v, final\_accuracy[i], round(final\_accuracy[i], 4),

              fontsize = 12,

              color='white',

              horizontalalignment='center',

              verticalalignment='bottom')

  plt.show()

print("maximum accuracy")

for key, value in maximum.items():

  print(key, value)