인공지능 1차 과제 – 컴퓨터과학부 2022920024 박동찬

1. Gaussian Distribution Binary Classification
   A. 문제 분석 및 설계

      주어진 Gaussian Dataset은 선형에 가깝게 분리가 가능한 형태이고 feature가 2개이며 label이 0, 1로 나누어져 있다. 따라서, Input node는 2개이고 output node는 1개로 구성된다. Neural Network를 구성할 때, hidden layer의 수, 각 hidden layer에 들어가는 node의 수, optimizer, activation function, loss function, learning rate, iteration, L1, L2 regularization를 다양하게 변경할 수 있다. 이것들을 적절히 조합하여 Test accuracy가 높은 값을 가지도록 만들어야 한다. Optimizer는 GD, Momentum, Adam 3가지를 바꾸어가며 적용한다. Activation function은 Linear와 ReLU, Sigmoid를 사용한다. Loss function은 Binary Classification에 자주 사용되는 BCE로 적용한다. (Linear의 경우, BCE 적용 못하므로 MSE 사용) Hyperparameter를 hidden layer 및 node 수, learning rate, iteration으로 정하고 Grid Search 방법을 통해 모델에 가장 적합한 hyperparameter를 찾는다. 또한 L1, L2 Regularization을 적용하여 L1, L2 lamda도 Grid Search 한다.

   B. 실험 (자세한 실험 결과는 코드 결과에 첨부, hidden에서 [8, 4, 2] 같은 형태는 node 8개, node 4개, node 2개를 3층 쌓는다는 의미)

      i. Optimizer: GD, Activation function: (hidden: ReLU, output: sigmoid), Loss: BCE
         Grid Search한 결과, hidden: [2], learning rate: 0.01, iterations: 5000일 때 Test Accuracy: 90%가 도출되었다.

      ii. Optimizer: Momentum, Activation function: (hidden: ReLU, output: sigmoid), Loss: BCE
         Grid Search한 결과, hidden: [8], learning rate: 0.1, iterations: 10000일 때 Test Accuracy: 90%가 도출되었다.

      iii. Optimizer: Adam, Activation function: (hidden: ReLU, output: sigmoid), Loss: BCE
         Grid Search한 결과, hidden: [4, 2], learning rate: 0.001, iterations: 20000일 때 Test Accuracy: 90%가 도출되었다.

      iv. Optimizer: GD, Activation function: sigmoid, Loss: BCE
         Grid Search한 결과, hidden: [2], learning rate: 0.01, iterations: 5000일 때 Test Accuracy: 90%가 도출되었다.

      v. Optimizer: GD, Activation function: linear, Loss: MSE
         Grid Search한 결과, learning rate: 0.001, iterations: 5000을 제외하고 모두 Test Accuracy: 90%가 도출되었다. (linear이므로 hidden layer 의미 없음)

      vi. Optimizer: GD, Activation function: (hidden: ReLU, output: sigmoid), Loss: BCE
         i 와 같은 실험 조건에서 Regularization option을 추가하여 Grid Search한 결과, hidden: [4], learning rate: 0.001, iterations: 5000, L1 lamda: 0.0, L2 lamda: 0.01일 때 Test Accuracy: 90.5%가 도출되었다.

   C. 분석

      Gaussian Distribution을 분류할 때 선형에 가깝게 분리되기 때문에 hidden layer를 많이 쌓는 구조는 대부분 Accuracy가 떨어졌다. 이 이유는 ReLU, Sigmoid를 적용한 경우, hidden layer를 많이 쌓을수록 비선형성이 강해져 제대로 분리하지 못했다. 위 실험에서 최대한 많은 경우를 다루려고 했으며 가장 좋은 모델이 대부분 accuracy=90% 정도로 나타났다. 위 결과들은 대부분 testset을 잘 구분해 냈지만, linear를 사용한 경우는 90%가 나타난 경우를 시각화했을 때 잘 구분해내지 못했다. 이는 Class imbalance 문제로 대부분이 특정 클래스에 속해 높은 정확도를 가지게 된 것으로 분석할 수 있고 MSE와 BCE와의 차이도 있을 것으로 보인다. Linear 모델에서 accuracy=90%가 나오지 않은 한 가지 경우는 Loss가 줄어드는 중에 학습 수가 부족해 Underfitting 되어 낮게 나온 것으로 분석했다. ReLU와 Sigmoid를 동시에 사용한 이유는 hidden layer에서 gradient vanishing을 완화하

고 계산을 빠르게 하기 위해 ReLU를 사용했고 output layer에서 binary classification을 수행하기에 유리한 sigmoid를 적용했다. 하지만 둘 다 사용한 구조와 sigmoid만을 사용한 구조의 결과값에 큰 차이점이 없었다. 가장 높은 정확도를 가진 모델은 실험 vi이며 Regularization이 overfitting을 방지하고 일반화 성능을 향상시킨 것으로 보인다. 위 실험에서 대부분의 경우의 loss 값이 증가했다는 점이 아쉬웠으며 이는 overfitting과 gradient explosion 등의 문제로 보인다. Leaky ReLU를 적용하거나 gradient clipping, early stopping 등을 적용하면 이를 방지할 수 있다.

2. Sprial Distribution Binary Classification

A. 문제 분석 및 설계

Spiral distribution은 비선형성이 강한 형태이므로 Neural Network를 구성할 때 hidden layer 및 node 수를 크게 늘리고 비선형 activation function을 사용해야 한다. 위 Gaussian 문제와 같이 Hyperparameter로 hidden layer, learning rate, iteration을 Grid Search로 가장 좋은 모델을 찾는다. Activation function은 hidden에서 ReLU, Leaky ReLU, tanh를 적용하고 output은 sigmoid를 사용한다. Optimizer는 위에서 사용한 것처럼 GD, Momentum, Adam으로 바꾸어가며 사용한다. 또한 Spiral 구조에서 feature를 그대로 쓰지 않고 극좌표로 변환해 사용하는 경우도 실험한다.

B. 실험 (자세한 실험 결과는 코드 결과에 첨부)

i. Optimizer: GD, Activation function: ReLU

Grid Search한 결과, hidden: [64, 32, 16], learning rate: 0.03, iterations: 3000일 때, Test Accuracy: 99.5%가 도출되었다.

ii. Optimizer: GD, Activation function: Leaky ReLU

Grid Search한 결과, hidden: [64, 32, 16], learning rate: 0.03, iterations: 3000일 때, Test Accuracy: 99%가 도출되었다.

iii. Optimizer: GD, Activation function: Tanh

Grid Search한 결과, hidden: [64, 32, 16], learning rate: 0.03, iterations: 3000일 때, Test Accuracy: 99%가 도출되었다.

iv. Optimizer: Momentum, Activation function: ReLU

Grid Search한 결과, hidden: [8, 6], learning rate: 0.05, iterations: 3000일 때, Test Accuracy: 97.5%가 도출되었다.

v. Optimizer: Momentum, Activation function: Leaky ReLU

Grid Search한 결과, hidden: [64, 32, 16], learning rate: 0.05, iterations: 3000일 때, Test Accuracy: 99%가 도출되었다.

vi. Optimizer: Momentum, Activation function: Tanh

Grid Search한 결과, hidden: [8, 8, 8, 8, 8], learning rate: 0.05, iterations: 1000일 때, Test Accuracy: 99%가 도출되었다.

vii. Optimizer: Adam, Activation function: ReLU

Grid Search한 결과, hidden: [64, 32, 16], learning rate: 0.03, iterations: 1500일 때, Test Accuracy: 99%가 도출되었다.

viii. Optimizer: Adam, Activation function: Leaky ReLU

Grid Search한 결과, hidden: [64, 32, 16], learning rate: 0.03, iterations: 1500일 때, Test Accuracy: 99%가 도출되었다.

ix. Optimizer: Adam, Activation function: Tanh

Grid Search한 결과, hidden: [64, 32, 16], learning rate: 0.03, iterations: 1000일 때, Test Accuracy: 99.5%가 도출되었다.

x. Optimizer: GD, Activation function: ReLU, Feature: Polar

i 와 같은 실험 조건에서 극좌표 변환을 적용하여 Grid Search한 결과, hidden: [8, 8, 8, 8, 8], learning rate: 0.03, iterations: 1500일 때, Test Accuracy: 98%가 도출되었다.

C. 분석

i. Hidden layer의 구조

위 실험에서 hidden layer를 [8, 6], [8, 8, 8, 8, 8], [64, 32, 16]로 Grid Search 했는데 대부분 [8, 6]일 때, Test Accuracy가 가장 낮았다. 특히 단순한 optimizer를 적용했을 때 작은 Neural network는 성능이 잘 나오지 않았다. 이는 비선형적으로 분류해야 하지만 hidden layer와 node 수가 부족해 비선형성이 강해지지 않았기 때문이다.

ii. Optimizer

GD를 사용한 실험은 높은 accuracy가 도출된 경우도 있었지만, 일반적으로 많은 iterations가 필요했고 learning rate에 따라 모델의 성능이 크게 바뀌었다. 높은 learning rate에서는 최적점을 지나쳐 진동하거나 수렴하지 않고 낮은 learning rate에서는 local minima 문제와 학습이 느려지게 되기 때문이다. 이를 통해 적절한 learning rate를 찾는 것이 중요하다는 것을 알 수 있다.

Momentum을 사용한 실험은 GD에 비해 빠르게 수렴하고 일부 실험에서 더 높은 성능을 보였다. 하지만 loss가 안정적으로 감소하지 않고 최적점 주변에서 불안정하게 수렴하는 overshooting의 문제가 나타났다.

Adam optimizer를 사용한 경우는 대부분의 실험에서 90% 이상을 유지하며 다른 optimizer에 비해 더 빠르고 안정적으로 학습했다. 또한 다른 optimizer와 비교했을 때 적은 iterations에도 적응형 학습률 조정을 통해 좋은 결과를 얻을 수 있었다.

iii. Activation function

ReLU와 Leaky ReLU는 높은 정확도를 유지하면서 학습 속도가 빠른 편이었다. 특히 깊고 큰 구조에서 최적의 성능을 보였다.

Tanh는 모든 실험에서 안정적인 성능을 보였으나 깊고 큰 구조에서 일부 경우는 학습이 불안정한 경우가 있었다. 또한 ReLU 계열에 비해 초기 수렴 속도가 느린 경향이 보였다. 하지만 Adam에서는 매우 높은 성능을 보였다.

iv. 극좌표 변환

극좌표 변환을 사용한 결과는 accruacy가 높아 보이지만 실제 시각화 결과는 좋지 않았다. 극좌표 변환으로 학습에 필요한 패턴이 왜곡되거나 더 복잡한 형태로 변환되면서 학습이 더 어려워졌을 가능성이 있다.

v. 실험 결과에서의 문제점

많은 실험 결과에서 학습에 따라 loss가 감소해야 하지만 오히려 증가하는 경우도 많았다. 이는 Learning rate가 너무 커서 overshooting 문제가 발생했거나 단순한 optimizer의 경우 가중치의 변화가 불안정해져 제대로 학습하지 못할 수 있다. 작성한 코드는 가중치를 초기화할 때 표준 난수로 초기화했는데 이때 부적절하게 초기화되면 loss가 오히려 증가할 수 있다. 이를 해결하기 위해 Xavier 초기화나 He 초기화를 적용하여 급격한 기울기 구간을 벗어나도록 조정할 수 있다. 또한 학습이 오래 진행되어 overfitting 되었을 수 있으므로 Early stopping 기법 등이 필요하다.

따라서 Spiral distribution에서 최적의 성능을 도출하기 위해서는 Adam optimizer와 적절히 큰 Neural Network 구조, ReLU 계열의 Activation function이 필요하고 적절한 hyperparameters를 Grid search하여 찾는 것이 유리하다. 또한 가중치 초기화 기법이나 Early stopping, Gradient clipping을 적용한다면 더 좋은 모델을 만들 수 있다.