Slide2

Learning rate가 크면 튕겨나가거나 숫자가 아닌 값이 나오는 overshooting이 발생하는데 프로그래밍 오류로 nan이 나옵니다. Learning rate가 너무 작으면 계산 과정이 너무 오래 걸리거나 cost function의 최저점이 아닌 곳에서 멈추게 됩니다. 그러므로 learning rate를 정할 때 0.01로 시작해서 cost function이 발산하면 learning rate를 더 작게, 계산 과정이 너무 오래 걸리면 learning rate를 좀 더 크게 하는 방식으로 learning rate를 조정합니다.

Slide3

두 데이터 값 차이가 크면 찌그러진 타원 모양이 나옵니다. 한 곳에서 learning을 시작하면 너무좁은 부분에서 너무 좁아서 밖으로 튀어나갈 수 있으므로 데이터 값을 표준화 해주어야 합니다. Learning rate를 제대로 설정하였는데 cost 함수가 발산하거나 이상이 있으면 preprocessing을 해주어야 합니다. Preprocessing을 거친 데이터는 주로 data의 중심이 0이 되는 zero-centered data와 데이터 값이 항 일정한 범위 안에 있도록 하는 normalized data가 됩니다. 이 표준화 식을 이용하면 데이터를 표준화할 수 있고 파이토치에서는 밑의 식을 이용하여 표준화할 수 있습니다.

Slide4

Overfitting은 model이 training set에 대한 답은 잘하지만 test set 또는 실생활에서는 잘 예측을 하지 못하는 경우인데 이는 linear하지 않게 training을 시킨 경우에 발생합니다. Overfitting을 예방하기 위해 training data의 양을 더 늘리거나 features의 수를 줄이거나 weight에서 너무 큰 수를 갖지 않게 하는 regularization을 합니다. Regularization은 구부러진 경계선을 linear하게 한다는 뜻입니다. Regularization을 하면 손실 함수에 이 뒷부분이 붙는데 이 람다는 regularization strength입니다. 오버피팅이 발생하였을 때 Regularization strength를 키울수록 오버피팅 구간을 벗어나게 됩니다. 하지만 너무 크면 언더피팅이 발생할 수 있으므로 조심해야 합니다.

Slide5

Data의 앞부분을 training set로, 뒷부분을 test set로 합니다. Training set를 training 시킨 후 test set으로 시험해 봐서 training이 잘 되었는지 확인합니다. Training set을 또 training 부분과 validation 부분으로 나눌 수 있는데 validation은 일단 training시키지 않고 training 부분만 training시킨 후 validation으로 시험해 보고 답이 틀리면 validation도 훈련시키는 방식으로 훈련시킵니다.

Online learning은 큰 데이터를 나누어서 순차적으로 학습하는 방식입니다. 초기교육단계가 완료되면 w와 b가 변경되지 않는 오프라인 learning과는 달리 w와 b를 계속업데이트할 수 있습니다.

Slide6

이 그림을 보면 or과 and는 linear하게 나눌 수 있어서 쉽게 학습시킬 수 있지만 xor의 경우 linear하게 나눌 수 없습니다. 실제로 한 개의 logistic regression unit으로는 xor을 학습시킬 수 없다는 것이 수학적으로 증명되었고 여러 개의 logistic regression unit을 이용하여 학습시킬 수 있다는 개념이 생겨났는데 이 것이 multiplayer percentions의 줄임말인 MLP입니다. 또한 거꾸로 계산하는 back propagation이 등장하여 좀 더 복잡한 데이터 학습, 예측이 가능해졌습니다. 하지만 몇몇 layer에서는 잘 작동하지만 수가 많아지면 앞의 error를 뒤로 보낼 때 error가 약해져서 잘 전달되지 않는다는 단점이 있습니다.

Slide7

Shape은 배열 안의 원소의 개수를 밖에서부터 안으로의 순서로 나타낸 것이고 rank는 배열의 차원, axis는 배열의 차원에 index를 매긴 것입니다. 배열을 곱할 때 곱하기 기호를 쓰는 것이 아니고 꼭 matmul을 써야합니다. 또한, 강의에서 위와 같은 것들을 소개하였고 자세한 내용은 생략하겠습니다.

Slide8

XOR문제는 앞에서 말했듯이 한 개의 logistic regression unit으로는 절대 풀 수 없고 세 개의 logistic regression unit으로 풀 수 있습니다. 강의에서 들었던 예시를 설명하면 x값 2개를 두 개의 unit에서 각각 계산하고 결과값을 받아서 마지막 unit에서 계산하는 형식입니다. W와 b값과 x1과 x2값을 이용하여 행렬 곱셈을 하고 그 결과를 sigmoid함수에 넣어서 결과값을 구합니다. 그 결과값 두개를 마지막 unit에서 같은 방식으로 계산하면 최종 결과값이 나옵니다. 첫번째 두 개의 unit을 하나로 합칠 수 있는데 이 때는 w와 b값을 각각 나란히 쓴 다음 방금과 똑같은 방법으로 계산하면 됩니다.

Slide9

Back propagation은 chain rule을 이용하여 기울기를 구하는 과정인데 저번 주에 파이토치 첫걸음 책 50p의 맨 위에 있는 그림 어떻게 푸는 건지 찾아냈으니 혹시 궁금하시면 좀 이따 질문 주시기 바랍니다.

Slide10

인공 신경망 모델을 사용하면 O와 X를 구분하는 모델에서 그냥 X와 기울어진 X를 전혀 다른 값으로 인식하기 때문에 성능이 잘 나오지 않는데 이 성능을 향상시키기 위해서 동물의 시각 뉴런에 착안한 합성곱 신경망이 등장하였습니다. 합성곱 신경망은 이미지가 들어오면 특정 영역에 대하여 합성곱 연산과 서브샘플링을 적용하는 것을 반복하고 나중에는 완전 연결 및 가우시안 연결을 사용하여 최종적인 결과값을 냅니다. 합성곱 연산 계산방법은 오른쪽 코드와 같고 활성화 지도의 크기는 이와 같은 공식으로 계산하고 이때 I는 이미지 크기, K는 필터 크기, S는 스트라이드입니다. 합성곱 연산은 비선형선을 추가하기 위해 활성화함수를 필요로 하는데 주로 렐루를 사용합니다.

Slide11

패딩은 더 깊은 신경망을 만들기 위해서 일정한 크기의 층으로 이미지를 감싸는 것입니다. 패딩을 적용하면 활성화지도 크기 공식이 이렇게 바뀝니다. 풀링은 입력 이미지가 엄청나게 크면 굳이 패딩을 적용하지 않아도 충분히 특성을 뽑아낼 수 있으므로 하는 것입니다. 맥스풀링과 평균 풀링이 있는데 맥스 풀링은 일정 크기의 구간에서 가장 큰 값만 전달하는 것이고 평균 풀링은 일정 크기의 구간 내의 평균을 전달하는 것입니다. 풀링 후에는 뽑은 특성들을 입력으로 받는 인공 신경망을 뒤에 붙여서 각 클래스 별 확률을 뽑아내거나 특정 수치들을 뽑아냅니다.(회귀)

Slide12

소프트맥스 함수는 신경망의 결과값을 함수로 바꿔줍니다. 공식은 위와 같습니다. 이제 이를 one-hot 벡터로 바꿔주어야 하는데 one-hot 벡터란 모든 벡터들 원소 중 하나만 1인 벡터를 말합니다. 모든 output과 각각 cross entropy loss를 계산해줍니다. Cross entropy loss 공식은 다음과 같고 일어날 확률이 작을수록 정보가 크고 일어날 확률이 클수록 정보가 적습니다. 밑에는 쿨백-라이블러 발산인데 이를 최소화하면 q와 p의 분포와 최대한 같아지게 할 수 있습니다. 교차 엔트로피 값은 예측이 잘못될수록 L1 손실보다 더 크게 증가하는데 그만큼 더 손실값이 크기 때문에 학습에서 사용하면 장점이 있습니다. 그래서 분류 문제에서는 교차 엔트로피 손실을 많이 사용합니다.

Slide13

유명한 모델들은 위와 같은 것들이 있고 ResNet이후로도 GoogLeNet과 ResNet의 장점을 합친 Inception v4, 잔차 연결을 더 촘촘히 한 DenseNet 등이 더 개발되었습니다.