

Lab-07-1 Tips

최대 가능도 추정(Maximum Likelihood Estimation)

- 주어진 데이터를 가장 잘 설명할 수 있는 모수(Parameter)를 추정하는 방법
→ 가능도 함수가 최대가 되는 지점의 모수를 MLE로 추정

- 경사상승(하강)법 : 극값을 찾기 위해 현재 지점의 경도의 반대 방향의 값을 더해주는 알고리즘
→ 이를 MLE 추정에 사용할 수 있음

→ 모수 추정에서 과적합의 문제가 발생할 수 있다

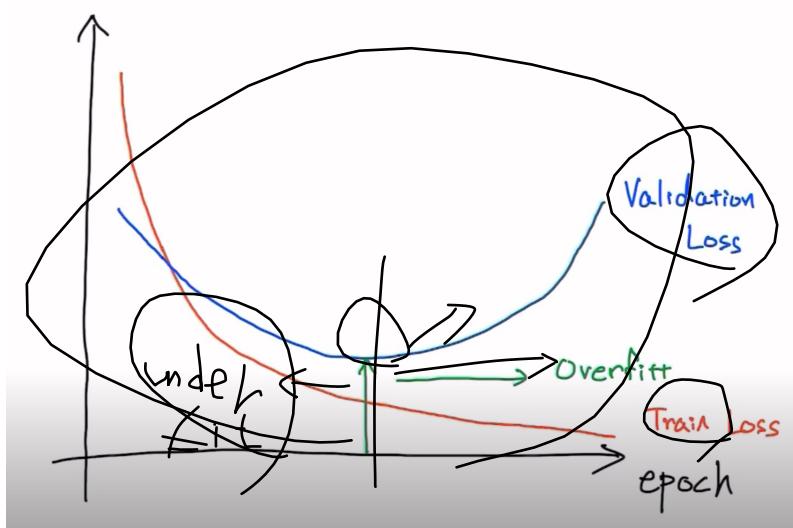
과적합(Overfitting)과 정규화(Regularization)

- Training Set의 데이터만을 설명할 수 있는 모수를 모집단의 실제 모수라고 잘못 추정하는 것
→ ex) 동전을 5번 던져서 모두 앞면이 나왔다고 앞면 나올 확률이 100%은 아니다



Development Set을 활용하여 과적합을 방지할 수 있음
(Dev Set 없이 Test Set 만 있는 경우도 있다)

- Validation Loss 가 가장 작은 epoch를 찾아서 최적의 학습을 진행한다
- 반대로 underfitting 이란 것도 있는데 이는 Train Loss 가 줄어들지 않는 경우이다



Overfitting

- More Data
- Less features : 모델의 복잡함과 성능은 비례하지 않는다
- **Regularization** : 특정 조건 하에서만 성립하는 모델 말고 일반화된 모델이 필요하다

Regularization

- Early Stopping : Validation Loss 가 더 이상 낮아지지 않을 때. 위의 그림이 바로 이거임
- Reducing Network Size
- Weight Decay
- Dropout
- Batch Normalization

파라미터나 신경망의 Size 를 제한하기

$$L(\beta) = \min_{\beta} \sum_{i=1}^n (y_i - \hat{y}_i)^2 + \lambda \sum_{j=1}^p \beta_j^2$$

(1) Training accuracy (2) Generalization accuracy

릿지추정량

불편추정량이 아니지만 MSE 가 감소함

Basic Approach to Train DNN

- ① Make a neural network architecture.
- ② Train and check that model is over-fitted.
 - a. If it is not, increase the model size (deeper and wider).
 - b. If it is, add regularization, such as drop-out, batch-normalization.
- ③ Repeat from step-2.

```

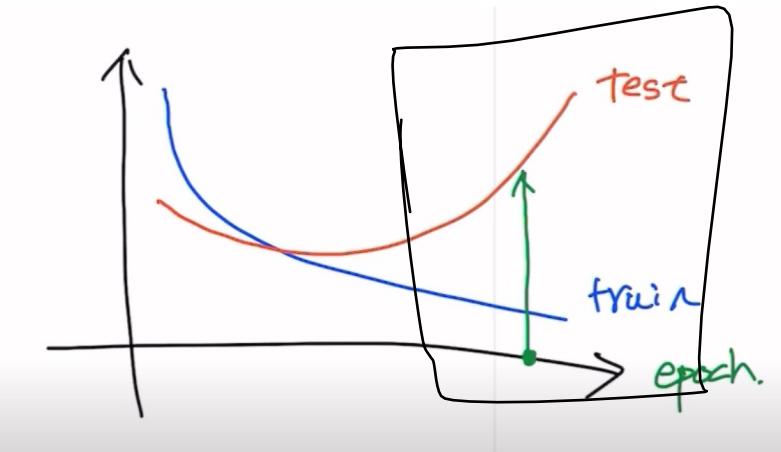
def test(model, optimizer, x_test, y_test):
    prediction = model(x_test)
    predicted_classes = prediction.max(1)[1]
    correct_count = (predicted_classes == y_test).sum().item()
    cost = F.cross_entropy(prediction, y_test)

    print('Accuracy: {}% Cost: {:.6f}'.format(
        correct_count / len(y_test) * 100, cost.item())
    )

```

test(model, optimizer, x_test, y_test)

Accuracy: 0.0% Cost: 1.425844



Data Preprocessing

$$x'_j = \frac{x_j - \mu_j}{\sigma_j}$$

여기서 σ 는 standard deviation, μ 는 평균값이다.

```
mu = x_train.mean(dim=0)
```

```
sigma = x_train.std(dim=0)
```

```
norm_x_train = (x_train - mu) / sigma
```

What is MNIST?

MNIST: handwritten digits dataset

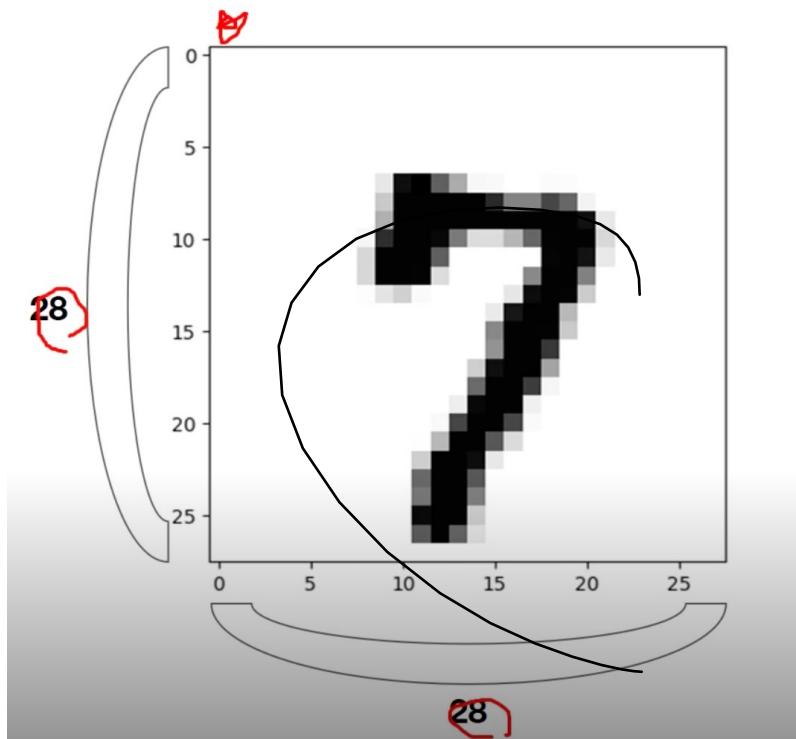


[train-images-idx3-ubyte.gz](#): training set images (9912422 bytes; 60,000 samples)

[train-labels-idx1-ubyte.gz](#): training set labels (28881 bytes)

[T10k-images-idx3-ubyte.gz](#) : test set images (1648877 bytes; 10,000 samples)

[T10k-labels-idx1-ubyte.gz](#) : test set labels (4542 bytes)



해상도 28x28 이미지
View 함수를 통해 이 이미지의
입력값 x 는 784 개의 데이터가 됨

torchvision

The torchvision package consists of popular datasets, model architectures, and common image transformations for computer vision.

Epoch / Batch size / Iteration

In the neural network terminology:

- one **epoch** = one forward pass and one backward pass of *all* the training examples
- **batch size** = the number of training examples in one forward/backward pass. The higher the batch size, the more memory space you'll need.
- number of **iterations** = number of passes, each pass using [batch size] number of examples. To be clear, one pass = one forward pass + one backward pass (we do not count the forward pass and backward pass as two different passes).