Idea Factory Intensive Program #2

딥러닝 홀로서기

#14

이론강의/PyTorch실습/코드리뷰

딥러닝(Deep Learning)에 관심이 있는 학생 발굴을 통한
딥러닝의 이론적 배경 강의 및 오픈소스 딥러닝 라이브러리 PyTorch를 활용한 실습

Today's Time Schedule

Assignment #1 Review

How to Parameterize Entire Code

How to Run Code with GPU!

1 hour?

How to Overcome Overfitting — 1 hour

Big Wave: Hyperparameter Tuning — 2 hour

Purpose of Hyperparameter Tuning?

Purpose of Hyperparameter Tuning?

Increase Model Performance

Purpose of Hyperparameter Tuning?

↓

Increase Model Performance

↓

Reduce True Risk(Generalization Error) of Model

Purpose of Hyperparameter Tuning?

↓

Increase Model Performance

↓

Reduce True Risk(Generalization Error) of Model

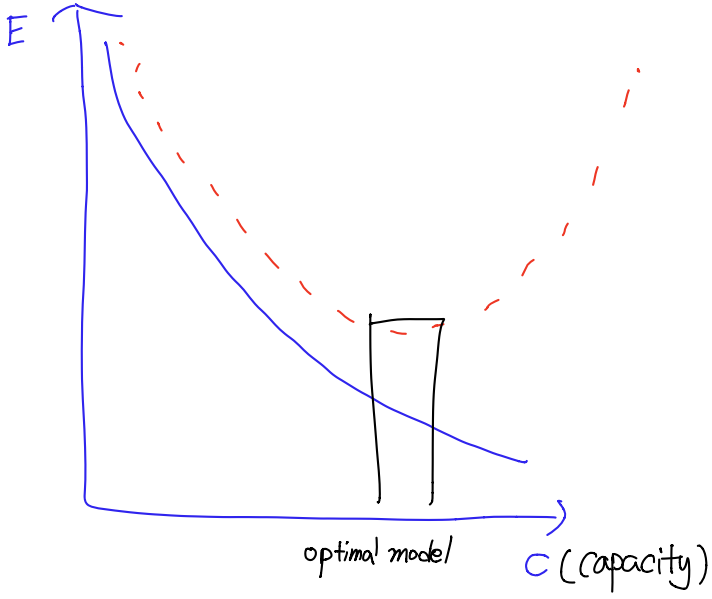
↓

Reduce True Risk on Validation Set, approximately

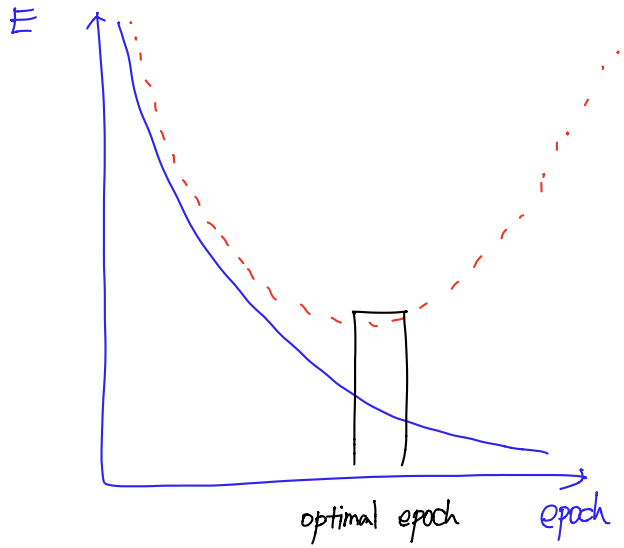
Two Approach of Hyperparameter Tuning

Options We Have for Hyperparameter Tuning

Model Related



Optimization Related



Options We Have for Hyperparameter Tuning

Model Related

Number of hidden layer

Number of hidden unit

Activation Function

Optimization Related

Type of Optimizer ← SGD. ADAM

Learning rate

L2 coef

Dropout Rate

Batch Size

Epoch

Options We Have for Hyperparameter Tuning

튜닝순서

Model Related

Number of hidden layer

Number of hidden unit

Activation Function

Optimization Related

Type of Optimizer

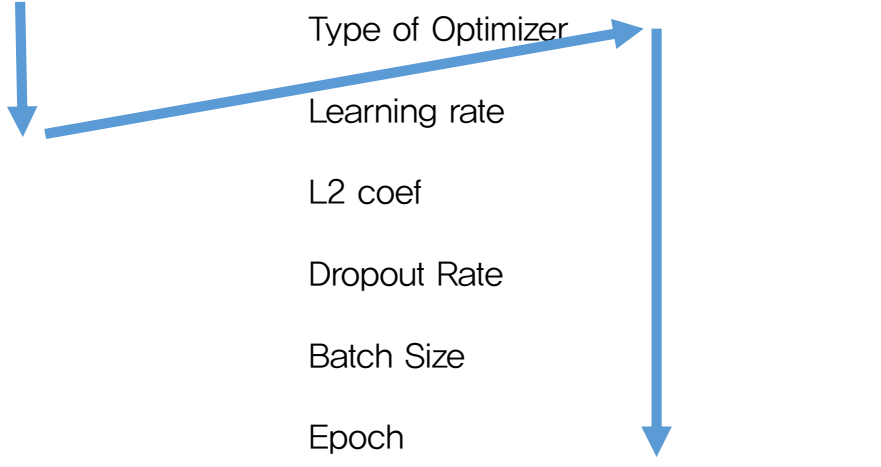
Learning rate

L2 coef

Dropout Rate

Batch Size

Epoch



Options We Have for Hyperparameter Tuning

Model Related

Number of hidden layer

Number of hidden unit

Activation Function

Very Flexible
(1∼10 layers for MLP) → 오~ 이정도 이상은 비효율적

(∼152, ∼1000 layers for Recent CNN)

Optimization Related

L2 coef

Dropout Rate

Batch Size

Epoch

Options We Have for Hyperparameter Tuning

Model Related

Number of hidden layer

Number of hidden unit

Activation Function

Very Flexible
(10~1024 unit/layer)

보통 2048개까지

Optimization Related

Type of Optimizer

g rate

L2 coef

Dropout Rate

Batch Size

Epoch

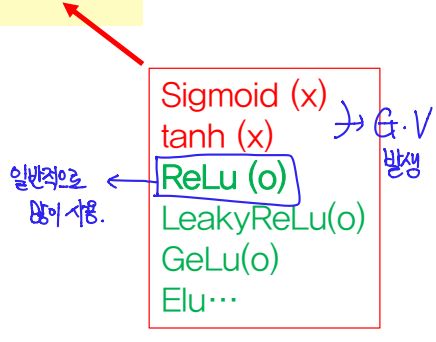
Options We Have for Hyperparameter Tuning

Model Related

Number of hidden layer

Number of hidden unit

Activation Function



Optimization Related

Type of Optimizer

Learning rate

L2 coef

Dropout Rate

Batch Size

Epoch

Options We Have for Hyperparameter Tuning

Model Related

Number of hidden layer

Number of hidden unit

Activation Function

Optimization Related

Type of Optimizer

Learning rate

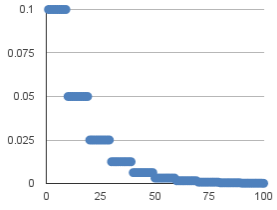
L2 coef

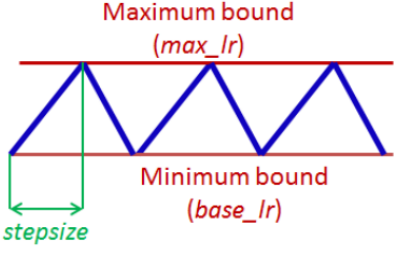
Dropout Rate

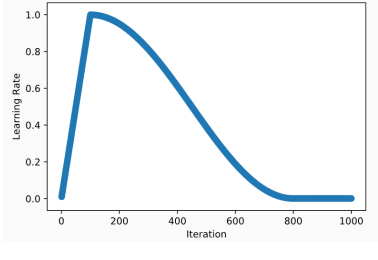
Batch Size

Epoch

Options We Have for Hyperparameter Tuning







Optimization Related

Learning Rate Scheduler

Type of Optimizer

Learning rate

L2 coef

Dropout Rate

Batch Size

Epoch

1e-5 ~ 1e-1
Log scale

0.00001
0.0001
0.001
0.01
0.1

0.001
0.003
0.006

Options We Have for Hyperparameter Tuning

Model Related

Number of hidden layer

Number of hidden unit

Activation Function

Optimization Related

Type of Optimizer

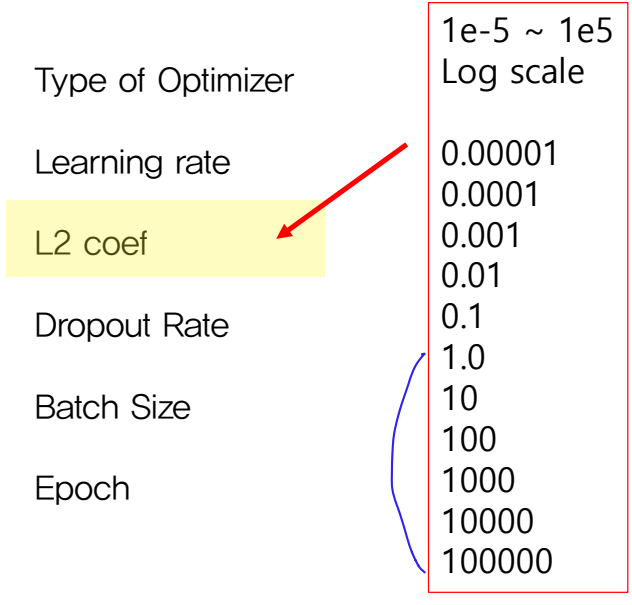
Learning rate

L2 coef

Dropout Rate

Batch Size

Epoch



Options We Have for Hyperparameter Tuning

Model Related

Number of hidden layer

Number of hidden unit

Activation Function

Optimization Related

Type of Optimizer

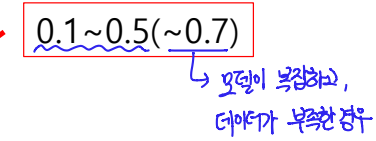
Learning rate

L2 coef

Dropout Rate

Batch Size

Epoch



Options We Have for Hyperparameter Tuning

Model Related

Number of hidden layer

Dependent on Data, Model

1) Increase batch size until OOM
2) If overfitting is severe, reduce batch size

Optimization Related

Type of Optimizer

Learning rate

L2 coef

Dropout Rate

Batch Size

Epoch

Options We Have for Hyperparameter Tuning

Model Related

Number of hidden layer

Regularly Measure Train Loss & Val Loss

Early Stopping
e.g. if validation loss is not reduced more than N epoch, then stop training

2~3 (tenacity)

Optimization Related

Type of Optimizer

Learning rate

L2 coef

Dropout Rate

Batch Size

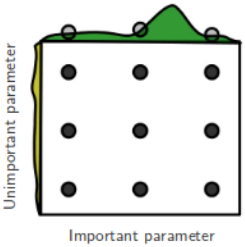
Epoch

Four Way to Tune Experiment

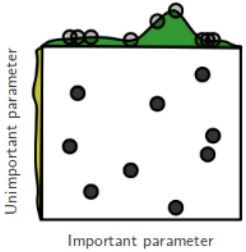
Four Way to Tune Experiment

⑤ Auto ML
ML을 ML로 학습시킴(재밌 바쁨.)

① Grid Layout



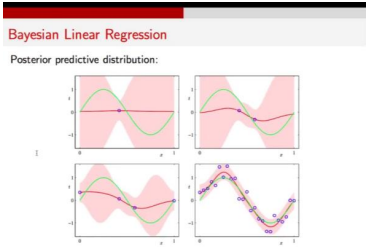
② Random Layout



③ Hand Tuning

밥 먹으며 10초에 한번 씩 Accuracy 체크하는 우리 내의 모습

④ Bayesian Optimization

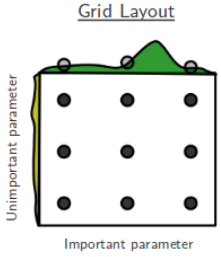


10개 랜덤으로 뽑고 Accuracy 측정
→ 이것을 바탕으로 그 근방의 값으로 최적화.

https://i.ytimg.com/vi/KhBB1J5JRzs/maxresdefault.jpg

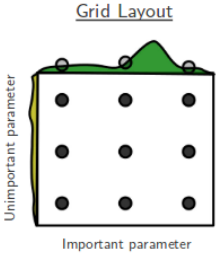
https://i.stack.imgur.com/cIDuR.png

Four Way to Tune Experiment

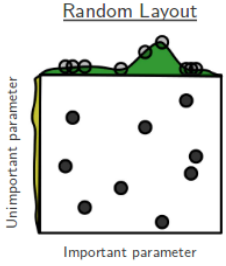


하이퍼파라미터에 따른
전체적인 경향성 파악

Four Way to Tune Experiment

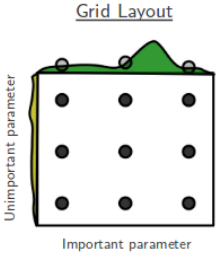


하이퍼파라미터에 따른
전체적인 경향성 파악

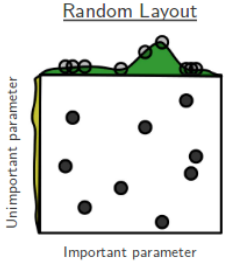


랜덤 서치로 미처 알지 못한 괜찮은 조합 탐색

Four Way to Tune Experiment



하이퍼파라미터에 따른
전체적인 경향성 파악

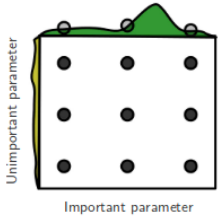


랜덤 서치로 미처 알지 못한 괜찮은 조합 탐색

(회사라면) 논문 읽는 척하면서 계속 하이퍼 파라미터 튜닝

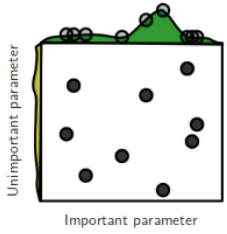
Four Way to Tune Experiment

Grid Layout



하이퍼파라미터에 따른
전체적인 경향성 파악

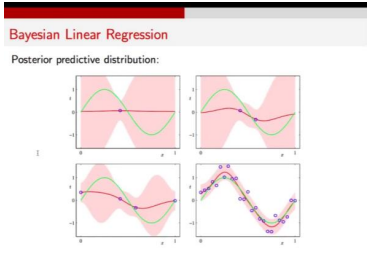
Random Layout



랜덤 서치로 미처 알지 못한 괜찮은 조합 탐색

(회사라면) 논문 읽는 척하면서 계속 하이퍼 파라미터 튜닝

Bayesian Optimization



어느 정도 파악이 되면 탐색하고 싶은 구간을
설정하고 오토 튜닝

Caution: Human Bias During Tuning Process

Caution: Human Bias During Tuning Process

1. Train / Validation / Test Set을 나누라고 시키길래 나누긴 나눔
2. 학습을 한번 돌려보고 Test Acc를 확인한다.
3. 그럭저럭 시작치곤 나쁘지 않다.
4. 하이퍼파라미터를 조금 바꿔보고 Test Acc와 Train/Val Loss 그래프를 확인해본다.
5. 오버피팅이 감지되면 고칠 만한 부분을 생각하고 해당 하이퍼파라미터를 바꿔본다.
6. 다시 Test Acc와 Train/Val Loss
7. 시행착오 끝에 Test Acc. 99% 달성~!

Caution: Human Bias During Tuning Process

1. Train / Validation / Test Set을 나누라고 시키길래 나누긴 나눔
2. 학습을 한번 돌려보고 Test Acc를 확인한다.
3. 그럭저럭 시작치곤 나쁘지 않다.
4. 하이퍼파라미터를 조금 바꿔보고 Test Acc와 Train/Val Loss 그래프를 확인해본다.
5. 오버피팅이 감지되면 고칠 만한 부분을 생각하고 해당 하이퍼파라미터를 바꿔본다.
6. 다시 Test Acc와 Train/Val Loss 를 모니터링
7. 시행착오 끝에 Test Acc. 99% 달성~!

Caution: Human Bias During Tuning Process

1. Train / Validation / Test Set을 나누라고 시키길래 나누긴 나눔
2. 학습을 한번 돌려보고 Test Acc를 확인한다.
3. 그럭저럭 시작치곤 나쁘지 않다.
4. 하이퍼파라미터를 조금 바꿔보고 Test Acc와 Train/Val Loss 그래프를 확인해본다.
5. 오버피팅이 감지되면 고칠 만한 부분을 생각하고 해당 하이퍼파라미터를 바꿔본다.
6. 다시 Test Acc와 Train/Val Loss 를 모니터링
7. 시행착오 끝에 Test Acc. 99% 달성~!

사람이 Test Set으로 학습되어 버림

→ 우리가 Test 셋에 hyperparameter를 맞춰버림.

Caution: Human Bias During Tuning Process

1. Train / Validation / Test Set을 나누라고 시키길래 나누긴 나눔
2. 학습을 한번 돌려보고 Test Acc를 확인한다.
3. 그럭저럭 시작치곤 나쁘지 않다.
4. 하이퍼파라미터를 조금 바꿔보고 ~~(Test Acc)~~ → (Val Acc)와 Train/Val Loss 그래프를 확인해본다.
5. 오버피팅이 감지되면 고칠 만한 부분을 생각하고 해당 하이퍼파라미터를 바꿔본다.
6. 다시 ~~(Test Acc)~~ → (Val Acc) 와 Train/Val Loss 를 모니터링
7. 시행착오 끝에 Test Acc. 99% 달성~!

Summary

Summary

1. MLP hidden layer 수가 바뀔 때마다 매번 코드에 직접 쳐야 함(개불편)
2. 실험 돌리는 거 오래 걸려요(현기증 남)
3. 같은 코드인데도 돌릴 때 마다 결과가 달라여?!?!?!
4. Train Loss는 줄어드는데 Validation Loss는 안 줄어들어요!
5. 변수들을 어떤 식으로 어떻게 바꿔야 할 지 모르겠어요!
6. 그리고 아직도 Train/Validation/Test 어떻게 써야 하는지도 모르겠어요!