

# DEEP LEARNING WITH PYTORCH

# MODEL TRAIN & TEST

# MODEL TRAIN & TEST

## ◆ 최적화 모듈 torch.optim

- W, b 업데이트 처리 클래스
- 필수 매개변수 : 모델 파라미터(model.parameters())와 학습률(lr)

유 형	특 징
torch.optim. <b>SGD</b>	가장 기본적인 최적화 알고리즘 각각의 파라미터에 대해 학습률(learning rate) 곱한 값 사용 가중치 업데이트
torch.optim. <b>Adam</b>	학습률을 각 파라미터마다 적응적으로 조절하는 최적화 알고리즘 현재 그래디언트와 이전 그래디언트의 지수 가중 평균 이용 가중치 업데이트
torch.optim. <b>RMSprop</b>	그래디언트의 제곱값의 이동 평균을 이용하여 학습률 조절하는 최적화 알고리즘 Adam과 유사한 방식으로 학습률 조절
torch.optim. <b>Adagrad</b>	각 파라미터에 대한 학습률을 조절하는 최적화 알고리즘 이전 그래디언트 제곱의 누적 값 사용하여 학습률 조절

---

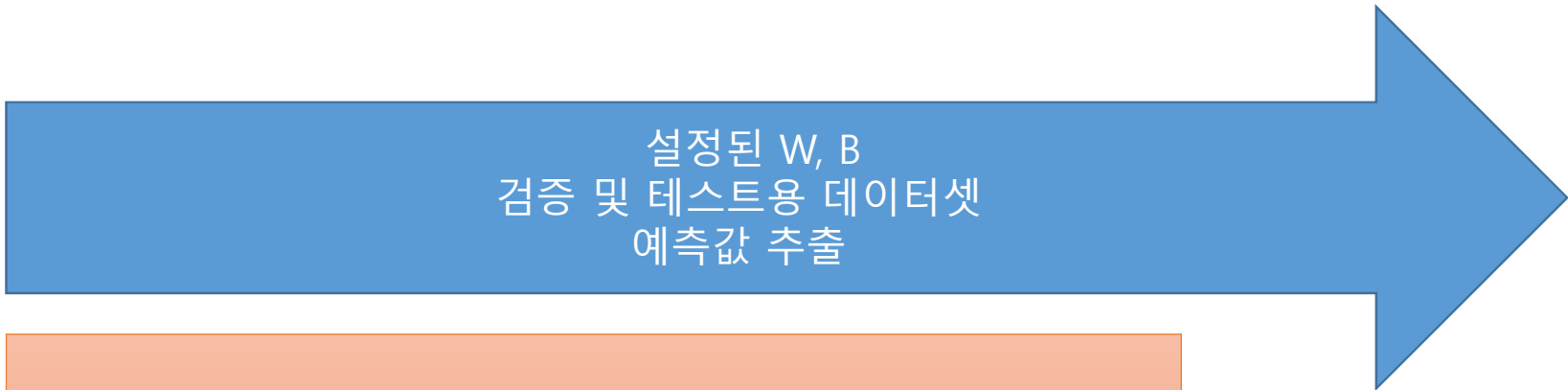
학습 상태

순방향

역전파 W,B 업데이트 : **Auto\_grade**

---

## 테스트 & 검증 상태



설정된 W, B  
검증 및 테스트용 데이터셋  
예측값 추출

\*\* 사용되지 않는 기능들 OFF

- W, B 업데이트 X

- ➔ 기능 OFF Auto\_grade 엔진 --- `model.eval()`

- ➔ W, B 텐서 `required_grad=True` --- `no_grad()`

# MODEL TRAIN & TEST

## ◆ 손실/비용 함수

- 모델 예측 값과 타겟 값의 차이 계산

유형		손실함수	
회귀		torch.nn. <b>MSELoss</b>	평균 제곱 오차(Mean Squared Error)
		torch.nn. <b>L1Loss</b>	평균 절대 오차(Mean Absolute Error)
분류	다중	torch.nn. <b>CrossEntropyLoss</b>	교차 엔트로피 손실(Cross Entropy Loss)
	이진	torch.nn. <b>BCELoss</b>	이진 교차 엔트로피 손실(Binary Cross Entropy Loss)

# MODEL TRAIN & TEST

## ◆ 모델 파라미터

- 모델의 학습 가능한 파라미터들을 반환하는 메소드

함수	역할
<b>model</b> <b>.parameters()</b>	<ul style="list-style-type: none"><li>• 모델의 학습 가능한 파라미터들 반환</li><li>• 모델의 모든 레이어 및 모듈에서 정의된 파라미터들 포함</li><li>• 모델이 학습 중에 업데이트되는 값들</li><li>• 최적화 알고리즘 통해 업데이트되는 대상</li><li>• 학습 데이터와 손실 함수를 통해 계산된 그라디언트를 사용하여 업데이트</li></ul>
<b>model</b> <b>.named_parameters()</b>	<ul style="list-style-type: none"><li>• parameters()와 동일 기능</li><li>• 파라미터 이름과 값을 전달</li></ul>

# MODEL TRAIN & TEST

## ◆ 모델 모드 설정

- 모델 구동에 따른 상태 설정

유형	설정 함수	
학 습 모 드	model.train()	<ul style="list-style-type: none"><li>* 모델 학습 전 호출 → 훈련 데이터에 대해 학습 시작할 준비</li><li>- 정규화(regularization) 기법들 동작 설정</li><li>- 모델 파라미터 업데이트를 위해 역전파(backpropagation) 수행</li><li>- 그래디언트 계산 수행</li><li>- 최적화(optimizer) 알고리즘에 따라 모델 파라미터 업데이트</li></ul>



# MODEL TRAIN & TEST

## ◆ 모델 모드 설정

- 모델 구동에 따른 상태 설정

유형	설정 함수	
평가 모드	model.eval()	<ul style="list-style-type: none"><li>* 모델 평가/추론 모드 전환</li><li>* 테스트 데이터나 검증 데이터 사용하여 모델 평가할 때 사용</li><li>- 정규화(regularization) 기법들 비활성화 설정</li><li>- 드롭아웃 비활성화, 배치 정규화 이동 평균/이동 분산 업데이트 않됨</li><li>- 역전파 비활성화로 모델 파라미터 업데이트 않됨</li></ul>

# MODEL TRAIN & TEST

---

## ◆ 모델 파라미터만 저장 & 로딩

- 학습한 매개변수를 `state_dict`라는 내부 상태 사전(internal state dictionary)에 저장
- 이 상태 값들은 `torch.save` 메소드 사용하여 저장(persist)

# MODEL TRAIN & TEST

## ◆ 모델 파라미터만 저장 & 로딩

### ❖ state\_dict

학습한 매개변수 즉, 내부 상태 저장 사전(internal state dictionary)

### ❖ 저장 : `model.state_dict()`

```
torch.save( model.state_dict(), 'model_weights.pth' )
```

### ❖ 로딩 : `model.load_state_dict()`

```
model.load_state_dict( torch.laod( 'model_weights.pth' ) )
```

# MODEL TRAIN & TEST

---

## ◆ 모델 구조 + 파라미터 저장 & 로딩

❖ 저장 : **torch.save()**

```
torch.save( model , 'model_weights.pth' )
```

❖ 로딩 : **torch.load()**

```
model = torch.laod( 'model_weights.pth' ) )
```

# MODEL TRAIN & TEST

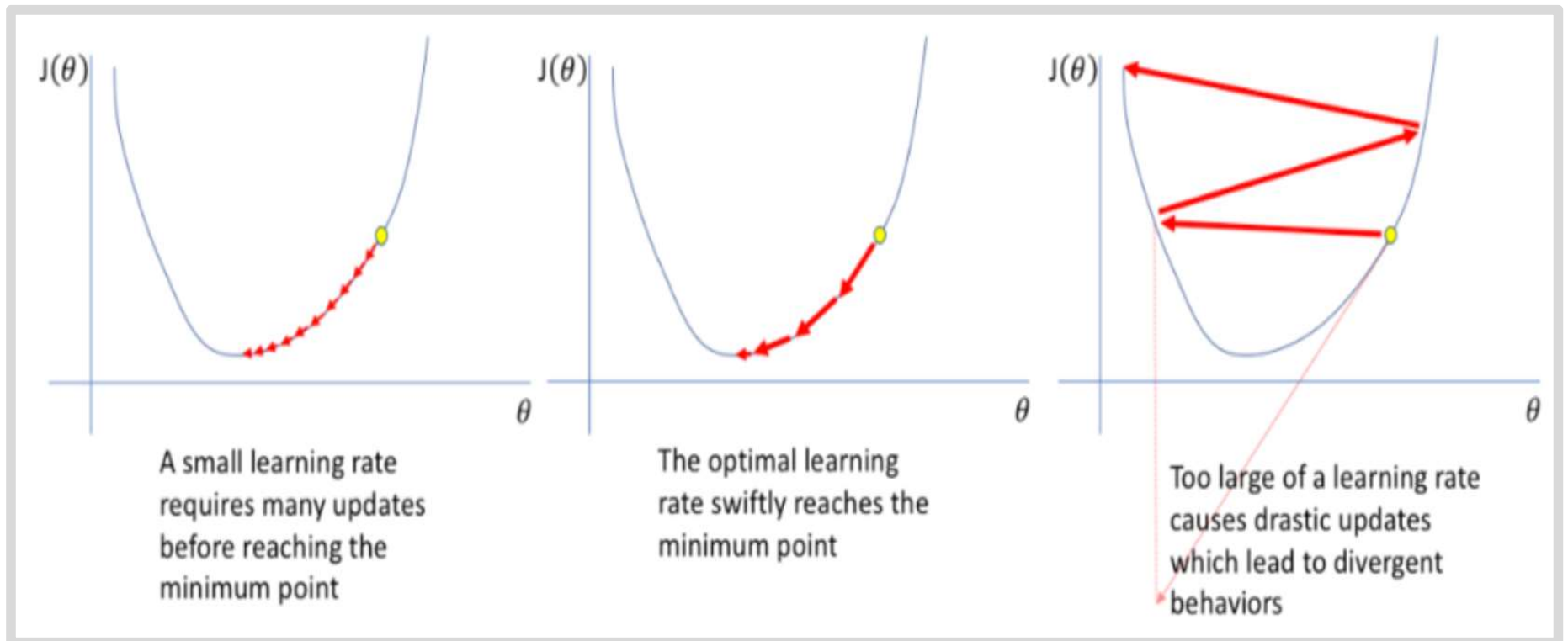
---

## ◆ 학습 스케줄러 Learning Rate scheduler

- 검증 손실이 더 이상 개선되지 않을 때 학습률 동적 감소 → 모델 학습 돕는 기법
- 학습 중에 주기적으로 검증 데이터셋의 손실 모니터링
- 미리 정의된 조건에 따라 학습률 감소
- 학습률 조절하여 **모델이 더 빠르게 수렴하며, 학습 과정에서의 안정성/성능 향상**

# MODEL TRAIN & TEST

## ◆ 학습 스케줄러 Learning Rate scheduler



# MODEL TRAIN & TEST

## ◆ 학습 스케줄러 Learning Rate scheduler

### ❖ Learning Rate Decay

- 기존 Learning Rate 높은 경우 : loss 값을 빠르게 내릴 수 있음 → 최적 학습 벗어날 수 있음
- 기존 Learning Rate 낮은 경우 : 최적 학습 가능 → 너무 오랜 시간 걸림



처음 시작시 Learning Rate 값을 크게 준 후 일정 epoch 마다 값을 감소  
최적 학습까지 더 빠르게 도달할 수 있게 하는 방법

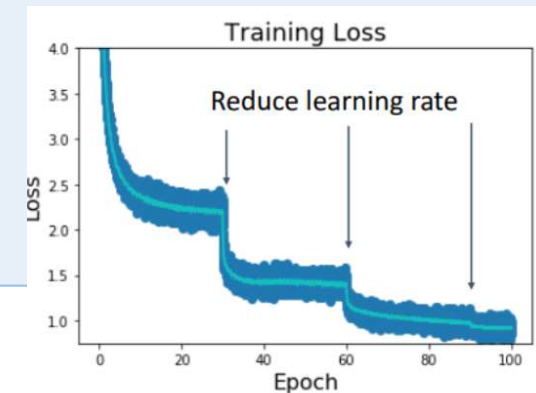
# MODEL TRAIN & TEST

## ◆ 학습 스케줄러 Learning Rate scheduler

### ❖ Learning Rate Decay

#### ▪ Step Decay

- 특정 epoch 구간(step) 마다 일정한 비율로 감소 시켜주는 방법
- 기존 Learning Rate에 추가 감소 비율(예제에선 0.1)
- 몇 epoch 구간(step)마다 감소를 시킬 것인지 설정
- learning rate 감소함에 따라 큰 폭으로 Loss가 0으로 수렴
- Loss가 연속적이지 않음





# MODEL TRAIN & TEST

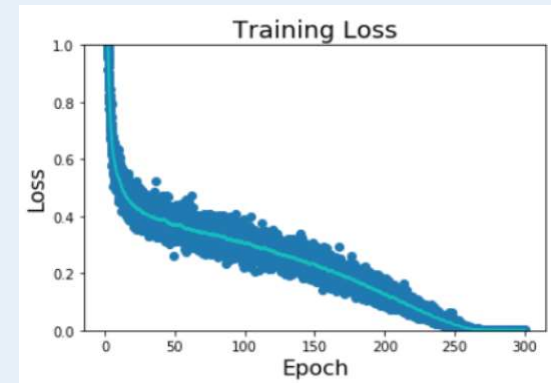
## ◆ 학습 스케줄러 Learning Rate scheduler

### ❖ Learning Rate Decay

#### ▪ Cosine Decay

- 안정적으로 끊임없이 loss가 감소
- Hyper Parameter 개수도 Step보다 적음

$$\text{Cosine: } \alpha_t = \frac{1}{2} \alpha_0 (1 + \cos(t\pi/T))$$



# MODEL TRAIN & TEST

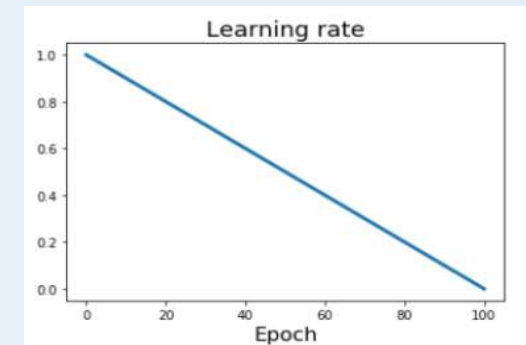
## ◆ 학습 스케줄러 Learning Rate scheduler

### ❖ Learning Rate Decay

#### ▪ Linear Decay

- 안정적으로 끊임없이 loss가 감소
- Hyper Parameter 개수는 Cosine과 동일
- 수식이 더욱 간단

**Linear:**  $\alpha_t = \alpha_0(1 - t/T)$



# MODEL TRAIN & TEST

## ◆ 학습 스케줄러 Learning Rate scheduler

### ❖ 스케줄러별 특징

StepLR	<ul style="list-style-type: none"><li>• step size마다 gamma 비율로 lr 감소<ul style="list-style-type: none"><li>→ step_size: 몇 epoch마다 lr 감소시킬지 의미</li><li>→ gamma: gamma 비율로 lr을 감소</li></ul></li></ul>
MultiStepLR	<ul style="list-style-type: none"><li>• learning rate를 감소시킬 epoch을 지정<ul style="list-style-type: none"><li>• milestones: learning rate 줄일 epoch index의 list</li><li>• gamma: gamma 비율로 lr을 감소</li></ul></li></ul>

# MODEL TRAIN & TEST

## ◆ 학습 스케줄러 Learning Rate scheduler

### ❖ 스케줄러별 특징

#### ReduceLROnPlateau

- 성능이 향상이 없을 때 learning rate를 감소
  - mode: 'min'이나 'max'중 하나 모드 설정
    - min - metric이 감소 멈출 때
    - max - metric이 증가 멈출 때
  - factor: 감소시킬 비율  $lr \times \text{factor}$  default: 0.1
  - patience: metric이 향상 안될 때, 몇 epoch을 참을 것인가?
  - threshold: 새로운 optimum이 될 수 있는 threshold

# MODEL TRAIN & TEST

## ◆ 학습 스케줄러 Learning Rate scheduler

### ❖ 스케줄러별 특징

#### ReduceLROnPlateau

- 성능이 향상이 없을 때 learning rate를 감소
  - threshold\_mode: dynamic threshold 설정 가능
    - 'rel'모드-  $\min \Rightarrow \text{best}(1 - \text{threshold})$   $\max \Rightarrow \text{best}(1 + \text{threshold})$
    - 'abs'모드-  $\text{best} + \text{threshold}$  무시
  - cool\_down: lr이 감소 후 몇 epoch동안 lr scheduler 정지
  - min\_lr: 최소 lr
  - eps: 줄이기 전, 줄인 후 lr의 차이가 eps보다 작으면 무시