Chatpor11.4~11.5

22.07.13

화학안전연구센터 최지원

11.4. 규제를 사용해 과대적합 피하기

- 심층 신경망은 전형적으로 수만 개에서 수백만 개의 파라미터를 가지고 있다.
- (+) 자유도가 높아 복잡한 데이터셋 학습 가능
- (-) 그만큼 네트워크의 자유도가 높기 때문에 과대적합되기에도 쉽다(규제의 필요성).

  [규제 방법]

  - 1) l_1 과 l_2 규제
  - 2) 드롭아웃
  - 3) 몬테 카를로 드롭아웃
  - 4) 맥스-노름 규제

11.4.1. l_1 과 l_2 규제(가중치 규제)

- 가중치 규제: 가중치의 값이 커지지 않도록 제한하는 기법
- 신경망의 연결 가중치를 제한하기 위해 l_2 규제를 사용하거나 희소 모델을 만들기 위해 l_1 규제를 사용할 수 있음
- l_1 규제: 손실함수에 가중치의 절대값(l_1)을 더한 값
- l_2 규제(가중치 감쇠): 손실함수에 가중치(l_2)의 제곱을 더한 값

$MSE(\text{손실함수}) + \alpha \cdot \text{L1norm}$ $\quad \|w\|_1 = \sum_{i=1}^{n} |w_i|$

l_1 norm

α	가중치	결과
↑	↓	underfitting
↓	↑	overfitting

$MSE(\text{손실함수}) + \gamma \cdot \text{L2norm}$ $\quad ||w_2|| = \sqrt{\sum_{i=1}^{n} |w_i|^2}$

l_2 norm

λ	가중치	결과
↑	↓ (기울기 감소, 특성의 영향 감소)	underfitting
↓	↑ (기울기 증가, 특성의 영향 증가)	overfitting

Ref. tistory.steadiness

11.4.1. l_1 과 l_2 규제(가중치 규제)

- 층마다 l_1 , l_2 혹은 $l_1 l_2$ 규제 지정하며, 기본 강도값은 0.01
- 일반적으로 모든 은닉층에 동일한 활성화함수, 초기화 전략, 규제방법을 공통적으로 적용
- 불필요한 반복을 피하기 위해 반복문을 사용하거나 functiontools.partial() 함수 사용

```
layer = keras.layers.Dense(100, activation="elu",
 kernel_initializer="he_normal",
 kernel_regularizer=keras.regularizers.l2(0.01))
```

dense 레이어(입출력을 각각 연결해주는 가중치), 활성화함수(ReLU 개선함수)
모델 성능 개선을 위한 가중치 초기화(He 초기화: 분산값을 지정해 그 범위 안에서 파라미터를 초기화)
가중치 규제 방식(e.g., l1, l2, l1_l2)

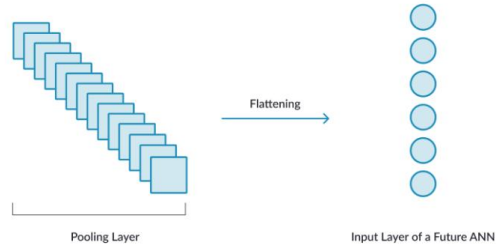
```
from functools import partial

RegularizedDense = partial(keras.layers.Dense,
 activation="elu",
 kernel_initializer="he_normal",
 kernel_regularizer=keras.regularizers.l2(0.01))

model = keras.models.Sequential([
 keras.layers.Flatten(input_shape=[28, 28]),
 RegularizedDense(300),
 RegularizedDense(100),
 RegularizedDense(10, activation="softmax",
 kernel_initializer="glorot_uniform")
])
```

동일하게 사용하는 기본적인 매개변수 값 사용 가능

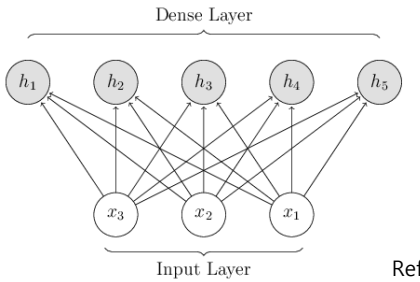
모델을 순차모델로 생성

추출된 주요 특징을 전결합층에 전달하기 위해 1차원으로 바꿔주는 layer

flatten layer



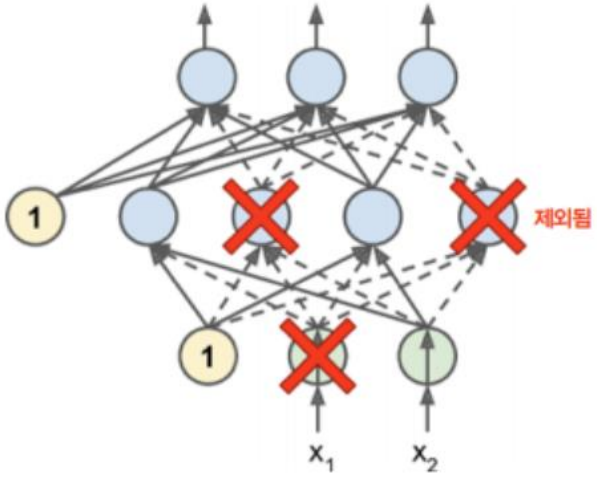
cf) dense layer



Ref. blog(yuns_u.log)

11.4.2. 드롭아웃

- 심층 신경망에서 가장 인기있는 규제 기법 중 하나이며, 훈련하는 동안 무작위로 층의 일부 출력 특성을 제외시킴
- 매 훈련 스텝에서 모든 뉴런들에 대해 일정 확률 p로 훈련에서 제외시키고 훈련한 나머지 뉴런들(=매 훈련마다 서로 다른 네트워크 가짐)을 평균한 앙상블로 볼 수 있음
  - 하이퍼파라미터인 드롭아웃 비율 p는 10~50% 사이
- 훈련 종료 후 예측, 테스트 시에는 드롭아웃없이 훈련 완료된 모델의 모든 뉴런들을 사용하여 예측 수행
- 모델이 과적합 되었을 경우 p를 높이고 과소적합되면 p를 낮춰야 함



11.4.2. 드롭아웃

- 드롭아웃은 훈련하는 동안에만 활성화되므로 훈련이 끝난 후 드롭아웃을 빼고 훈련 손실을 평가해야 함
- 드롭아웃 비율로 0.2를 사용한 드롭아웃 규제를 모든 dense 층 이전에 적용하는 코드는 아래와 같음

```
model = keras.models.Sequential([
 keras.layers.Flatten(input_shape=[28, 28]),
 keras.layers.Dropout(rate=0.2),
 keras.layers.Dense(300, activation="elu", kernel_initializer="he_normal"),
 keras.layers.Dropout(rate=0.2),
 keras.layers.Dense(100, activation="elu", kernel_initializer="he_normal"),
 keras.layers.Dropout(rate=0.2),
 keras.layers.Dense(10, activation="softmax")
])
```

모든 뉴런 중 20% 만큼 뉴런을 훈련에서 제외시키고 80% 뉴련이 모델 훈련에 참여

11.4.3. 몬테 카를로 드롭아웃

- 훈련 및 테스트 과정에서 모두 드롭아웃을 적용하는 방법
- 하나의 테스트 샘플에 대해 서로 다른 n번(사용자 지정 몬테카를로 샘플 수)의 예측 수행
- **랜덤성 있는 예측**(n개의 서로 다른 예측값)으로부터 **예측의 불확실성을 확인** 할 수 있는 통계량(e.g., 표준편차)를 구하는게 가능

몬테카를로 드롭아웃

```
y_probas = np.stack([model(X_test_scaled, training=True)
 for sample in range(100)])
y_proba = y_probas.mean(axis=0)
```

training=True (dropout 층 활성)
테스트 세트 샘플 수: 10,000개, 클래스 수: 10개 -> [1000,10] 모양의 행렬 100개
[100, 10000, 10] 모양의 행렬이 y_probas에 저장됨

y_probas (dropout으로 예측한 값들의 평균값)

모델 훈련 중 다르게 작동하는 층을 가지는 경우

```
class MCDropout(keras.layers.Dropout):
 def call(self, inputs):
 return super().call(inputs, training=True)
```

dropout층을 MCdropout 클래스로 변경

11.4.4. 맥스-노름 규제

- 각각의 뉴런에 대해 입력의 연결 가중치 w가 $||w||_2 \leq r$ 이 되도록 제한.
- r은 맥스-노름 하이퍼파라미터이며, $||\cdot||_2$는 l_2 노름을 나타냄.
- 매 훈련 스텝 종료 시 w의 l_2 노름을 계산하고 w의 스케일을 조정함.
- r을 줄이면 과대적합을 감소 시킬 수 있음.

$$w \leftarrow w \frac{r}{||w||_2}$$

맥스-노름 규제

```
keras.layers.Dense(100, activation="elu", kernel_initializer="he_normal",
 kernel_constraint=keras.constraints.max_norm(1.))
```

가중치들의 norm값이 최대 1로 제한시킴

11.5. 요약 및 실용적인 가이드라인

- 하이퍼파라미터 튜닝을 크게 하지 않고 대부분 경우에서 잘 맞는 기본 설정은 아래와 같음.
- 네트워크가 완전 연결 층을 쌓은 단순 모델이라면 자기 정규화(self-normalize)를 사용할 수 있음
- 아래의 가이드라인에서도 예외적인 경우가 존재함(11.5 참고)

기본 DNN 설정

하이퍼파라미터	기본값
커널 초기화	He 초기화
활성화 함수	ELU
정규화	얇은 신경망일 경우 없음, 깊은 싱경망이면 배치 정규화
규제	조기 종료 (필요하면 l2 규제 추가)
옵티마이저	모멘텀 최적화 (또는 RMSProp이나 Nadam)
학습률 스케줄	1 사이클

자기 정규화를 위한 DNN 설정

하이퍼파라미터	기본값
커널 초기화	르쿤 초기화
활성화 함수	SELU
정규화	없음 (자기 정규화)
규제	필요하다면 알파 드롭아웃
옵티마이저	모멘텀 최적화 (또는 RMSProp이나 Nadam)
학습률 스케줄	1 사이클