CHEAT SHEET

Python Code

[딥러닝으로 할 수 있는 것]

- 이미지 처리:
- 이미지 분류:

1. 지도학습: 사전에 주어진 정답 데이터를 바탕으로 학습하는 머신러닝 방법
   (레이블을 추정하는 분류 문제와 연속 값을 추정하는 회귀 문제 등이 있음)
2. 전이학습: 어떤 문제를 풀기 위해서 학습된 모델을 이용해 다른 문제를 풀 모델을 구축하는 방법
3. 원샷학습(one-shot learning): 1개 또는 아주 적은 샘플만을 가지고 학습하는 머신러닝 방법의 하나

- 강화 학습: 게임 공략, 시스템 제어

DQN(Deep Q Network, 심층강화학습): 딥러닝과 강화학습의 일종인 Q-Learning을 조합한 방법
= 딥러닝과 강화학습을 조합하는 발상은 이전부터 있었지만, 학습을 안정화 시키지 못함

- 시계열 데이터의 예측 · 분류: 주가 예측, 행동 추정 연구

- 이상 감지: 신용카드의 부정 감지, 시스템의 고장 감지, 불량품의 검출

많은 데이터와는 움직임이 다른 데이터를 검출하는 기술

Python Code

[Dense Layer]

1. 객체 생성 후 다시 호출하면서 입력 값 설정

```
dense = tf.keras.layers.Dense(...)
output = dense(input)
```

2. 객체 생성 시 입력값 설정

```
output = tf.keras.layers.Dense(...)(input)
```

3. dense layer 내장 옵션

```
_init_(
units, 출력 값의 크기, Integer 혹은 Long 형태
activation = None, 활성화 함수
use_bias = True, 편향(b)을 사용할지 여부
kernel_initializer = 'glorot_uniform', 가중치(W) 초기화 함수
bias_initializer = 'zeros', 편향 초기화 함수
kernel_regularizer=None, 가중치 정규화 방법
bias_regularizer=None, 편향 정규화 방법
activity_regularizer=None, 출력 값 정규화 방법
kernel_constraint=None, Optimizer에 의해 업데이트된 이후에 가중치에 적용되는 부가적인 제약함수
bias_constraint-None, Optimizer에 의해 업데이트된 이후에 편향에 적용되는 부가적인 제약 함수
**kwargs
)
```

[Dropout]:

과적합 문제는 정규화(Regularization) 방법을 사용해서 해결하는데, 그중 가장 대표적인 방법이 드롭아웃(dropout)이다.

1. 객체 생성 후 다시 호출하면서 입력값 설정

```
dropout = tf.keras.layers.Dropout(...)
Output = dropout(input)
```

2. 객체 생성 시 입력값 설정

```
Output = tf.keras.layers.Dropout(...)(input)
```

3. Dropout 내장 옵션

```
_init_(
rate, 드롭아웃을 적용할 확률을 지정(확률 값이므로 0~1 사이 값을 받음.
      예를 들어 dropout = 0.2로 지정하면 전체 입력값 중 20%를 0으로 만듦)
noise_shape=None, 정수형의 1D-tensor 값을 받음. 여기서 받는 값은
                  shape을 뜻함. 예를 들어 입력 값이 이미지일 때
                  noise_shape을 지정하면 특정 채널에만 드롭아웃을 적용할 수 있음.
seed=None, seed 값은 정수형이며, 같은 seed 값을 가지는 드롭아웃의 경우
           동일한 드롭아웃 결과를 만듦
**kwargs
)
```

= 드롭 아웃은 여러 층에 범용적으로 사용되어 과적합을 방지하기 때문에 모델 구현에 자주 사용하는 기법