**2021.01.09 10시~13시 1차시 스터디 복습**

-부려먹기 좋은 능력을 가진 사람이 되자

머신러닝의 기본 개념을 알고 그 알고리즘을 설명할 수 있는 사람이 되자

2. 자연어처리 개발 준비

1) 텐서플로

텐서는 n차원의 매트릭스(행렬)을 의미하며 말 그대로 텐서를 flow한다는 것은 흐름 그래프를 사용해 수치연산을 하는 과정을 의미한다.

(1) tf.keras.layers

텐서플로를 이용해 하나의 딥러닝 모델을 만드는 것은 마치 블록을 하나씩 쌓아서 전체 구조를 만들어 가는 것과 비슷하다. 따라서, 쉽게 블록을 바꾸고 여러 블록들의 조합을 쉽게 만들 수 있다는 것은 텐서플로의 큰 장점이다.

--------tf.keras.layers.Dense-----------------

Dense는 신경망 구조의 가장 기본적인 형태 즉

y=f(Wx+b) 수식을 만족하는 기본적인 신경망 형태의 층을 만드는 함수

x, b=입력벡터

W=가중치 행렬

가중치와 입력벡터를 곱한 후 편향을 더해준다. 그 값에 활성화 함수를 적용하는 구조.

가중치와 입력벡터를 곱한 후 편향을 더해주고 그 값에 활성화 함수를 적용해 임계치를 넘으면 1, 넘지 못하면 0을 도출한다.

**활성화 함수의 예시: 계단함수, 로지스틱 시그모이드, 하이퍼볼릭 탄젠트, 소프트플러스, 렉티파이어(ReLU)**

왼쪽 노드(점)이 입력값인 x가 되고 오른쪽 노드들이 y가 된다. 그리고 중간에 있는 선(에지)이 가중치를 곱하는 과정. 곱해지는 가중치들이 W.

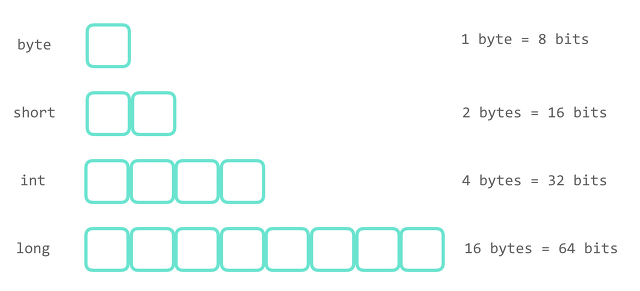
W와 b를 각각 변수로 선언한 후 행렬곱을 통해 구함(행렬로 변환해 계산)

케라스의 Dense를 이용하기 위해서는 먼저 객체를 생성해야한다.

생성한 dense층 객체에 입력값을 넣어야한다. 입력값을 넣기 위해서는 객체를 생성할 때 함께 넣거나 생성한 후 따로 적용하는 방법이 있다.

\*\*\*\*\*\*\*Dense층을 만들 때 정할 수 있는 옵션(객체를 생성할 때 지정할 수 있는 인자)\*\*\*\*\*\*\*

\*unit: 출력값의 크기 integer 혹은 long형태



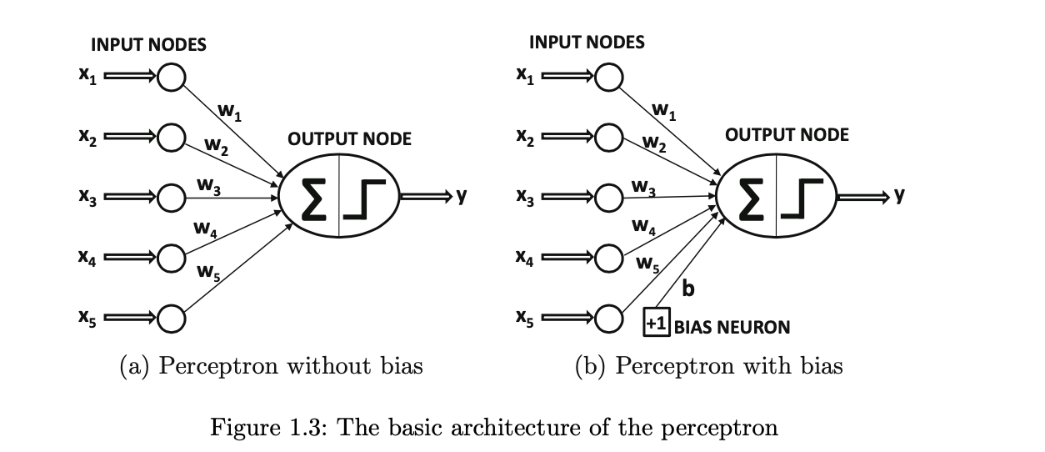
long형태는 64bit

\*activation: 활성화 함수. 가중치와 입력변수를 곱한값에 편향을 더한 뒤 그 값에 활성화 함수를 적용해 임계치를 넘어가면 1(예시), 넘어가지 않으면 0(예시)을 도출한다.

텍스트이(가) 표시된 사진

자동 생성된 설명

\*use\_bias: 편향(b)를 사용할지 여부. Boolean값 형태



Bias: 편향, 한쪽으로 쏠리는 것. 추가적으로 입력하는 엣지. bias함수의 진면목은 instagram필터에 있다. 이미지인식은 어차피 숫자로 되어있으니 특정값을 편향을 통해 턱을 깎거나 눈을 키우거나 할 수 있다.

Boolean값: 참/거짓 형태의 논리 자료형, TRUE/FALSE형태로 표시된다.

주로 참은 1, 거짓은 0에 대응하나 언어마다 차이가 있다. 숫자를 쓰지 않고 참과 거짓을 나타내는 영단어 true와 false를 쓰기도 한다. 불리언(Boolean)이라는 말은 영국의 수학자 겸 논리학자인 [조지 불](https://ko.wikipedia.org/wiki/%EC%A1%B0%EC%A7%80_%EB%B6%88)(George Boole)의 이름에서 따온 것이다.

따라서, use\_bias=True/False형태로 표시된다.

\*kernel\_initializer: 가중치 초기화함수

\*bias\_initializer: 편향 초기화함수

\*kernel regularizer: 가중치 정규화 방법

kernel: 가중치(통계)

정규화: regularization, 벡터의 사이즈/크기(norm)를 최소화 하는 것

~~(정규화는 정민화의 정도 아니고 벨타입 어쩌구도 아니고.. 벡터의 크기를 최소화하는 것)~~

? 가중치가 kernel이면 weight는? Weight가 들어간 함수가 kernel함수임.

\*bias\_regularizer: 편향 정규화 방법

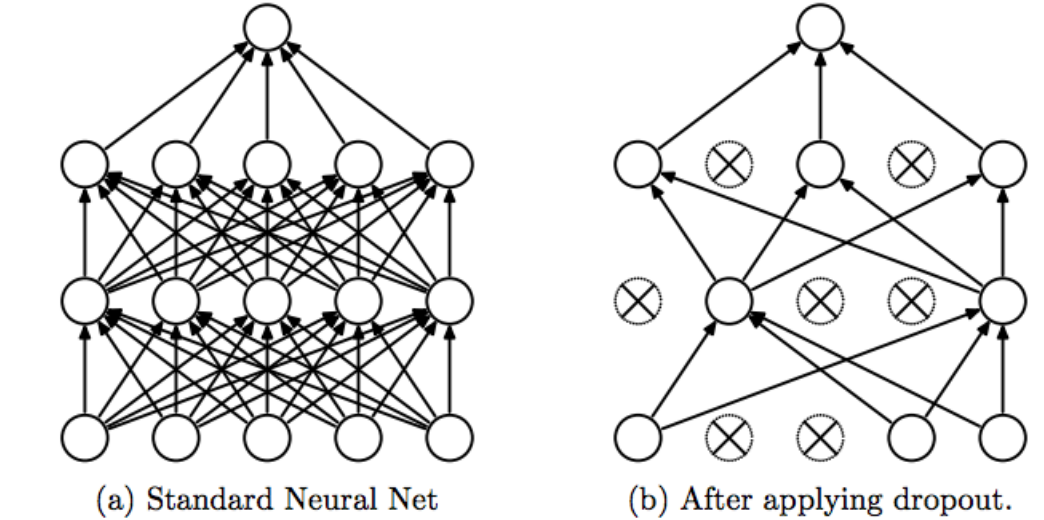
\*activity\_regularizer: 출력 값 정규화 방법

\*kernel\_constarint: optimizer에 의해 업데이트 된 이후에 가중치에 적용되는 부가적인 제약함수. 가중치를 업데이트 한 뒤에 적용되는 한도.(이 범위 안에서 알아서 해결햇!)

\*bias\_constraint: optimizer에 의해 업데이트 된 이후에 편향에 적용되는 부가적인 제약함수

--------tf.keras.layers.Dropout-----------------

과적합을 해결하기 위해 정규화 방법을 사용해서 해결하는데 그 중 대표적인 방법이 드롭아웃이다. 드롭아웃이란 필요없는 것들을 빼내는 것을 의미한다. 예)unit을 제거한 경우



\*\*\*\*\*\*\*\*\*\*\*\*\*드롭아웃 객체에서 설정할 수 있는 인자들\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*rate: 확률. 드롭아웃을 지정할 확률을 지정. 0~1사이의 값. 예를 들어 dropout=0.2이면 20%를 날린다.

\*noise\_shape: 정수형의 1dimension-tensor값을 받는다. 이때 받은 값(shape)을 지정함으로써 특정 값만 드롭아웃을 적용할 수 있다.

Shape: 형상. 데이터를 표현하는 숫자들

|  |
| --- |
| 예)비디오 데이터  - 비디오는 이미지에 프레임 개념이 추가된다.  - 하나의 비디오는 프레임의 연속, 각 프레임은 컬러이미지  - (샘플비디오수, 프레임수, 높이, 너비, 컬러채널수)의 형태  - 예를 들어, 60초짜리 120x140 영상을 초당 4프레임으로 샘플링하면 240프레임  - 이런 영상이 5개라면, (5, 240, 120, 140, 3) shape을 띔. |

예를 들어, 정민화라는 사람을 표현하려면 (25(나이), 0(female), 201803250(학번), 0402(생일), 10(눈썹형태))등등으로 표현할 수 있다. 이때 이미지인식에서 나를 인식할 땐 성별정보는 중요하지만 학번이나 생일 정보는 중요하지 않다. 이때 학번, 생일 등의 정보를 드롭아웃을 통해 제거한다.

\*seed: 무작위. 지정된 확률 값을 바탕으로 무작위로 드롭아웃을 적용하는데 이때 임의의 선택을 위한 시드값을 의미한다. seed값은 정수형, 같은 seed값을 가지는 드롭아웃의 경우 동일한 드롭아웃 결과를 만듦

tf.keras.layers.dropout의 rate=0.2면 20%날림

tf.nn.drpout의 rate=0.2면 80%날림(tf.nn모듈에서 사용함)

------------------tf.keras.layers.Conv1D----------------------

참고: <https://wikidocs.net/80437>

합성곱이란? 컴퓨터는 이미지를 하나의 큰 그림으로 인식하지 못함 그래서 필터의 크기만큼으로데이터를 잘라서 가로 방향으로 옮겨가면서 입력값에 대한 합성곱을 수행함. 연산 결과들이 모여 최종 출력값이 나옴

필터=커널

자연어 처리 분야에서 합성곱은 각 단어(혹은 문자) 벡터의 차원 전체에 대해 필터를 적용시키기 위해 주로 Conv1D를 사용한다.

Conv1D에서 (5,10)형태의 입력값에 대해 필터의크기인 kernel\_size=2로 설정하고 필터의 개수를 10으로 지정할 경우 출력값의 형태는 (1,4,10)이 된다.=1차원, 4행,

\*filters: 필터(커널)의 개수. 정수형, 출력의 차원수 나타냄

\*kernel\_size: 필터의 크기. 정수, 정수의 리스트, 튜플. 합성곱이 적용되는 윈도우의 길이.

둘 다 복수개의 값을 담을 수 있는 데이터 구조  
list - mutable (생성된 후에 변경 가능)  
tuple - immutable (생성된 후에 변경 불가능)

\*strides: 적용할 스트라이드의 값. 필터를 돌릴 횟수. 정수 혹은 정수의 리스트, 튜플형태로 저장. 1이 아닌 값 지정할 경우 dilaton\_rate는 1이외의 값 지정하지 못한다.

\*padding: 이미지 외곽에 의미 없는 경계를 덧붙이는 것. 가장자리 인식이 떨어질 때 사용함. ㅍVALID또는 SAME을 지정할 수 있다.

계속 합성곱 연산을 학하게 되면, 이미지가 계속 축소 됩니다.

가장자리 픽셀은 단 한번만 사용됩니다. 즉, 덜 사용됨으로서 이미지 윤곽쪽의 정보를 버리게 됩니다. 패딩은 이미지 주위에 추가로 하나의 경계를 덧대는 것입니다.

\*data\_format: 데이터의 표현방법.

\*dilation\_rate: dilation은 팽창. 합성곱 사용 시에 확장된 layer를 하나 추가하는 것. 1이 아닌 값을 지정하면 strides값으로 1이외의 값을 지정하지 못한다.

----------------tf.keras.layers.MaxPool1D-------------------

합성곱 신경만과 함께 쓰이는 방법은 풀링. 피처맵(입력으로부터 커널을 사용하여 합성곱 연산을 통해 나온 결과값(행렬))의 크기를 줄이거나 주요한 특징을 뽑아내기 위해 합성곱 이후에 적용되는 기법.

맥스풀링: 피처맵에 대해 최댓값만을 뽑아내는 방식

화살이(가) 표시된 사진

자동 생성된 설명

평균 풀링: 전체 값들을 평균한 값을 뽑는 방식

화살이(가) 표시된 사진

자동 생성된 설명

\*\*\*\*\*\*\*\*맥스풀링에 설정할 수 있는 인자\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*pool\_size: 풀링을 적용할 필터의 크기. 정숫값

\*strides: 정수 혹은 None

\*padding: 패딩방법. Valid 혹은 Same

\*data\_format: 데이터의 표현방법.

**2021.01.10 2장**

2) Tensorflow 2.0

API란?

**API**(Application Programming Interface, 응용 프로그램 프로그래밍 인터페이스)는 응용 프로그램에서 사용할 수 있도록, 운영 체제나 프로그래밍 언어가 제공하는 기능을 제어할 수 있게 만든 인터페이스(컴퓨터 및 소프트웨어 조작방법)를 뜻한다.

프로그램들이 서로 상호작용하는 매개체

자연어처리를 예로 하면 데이터가 있으면 그걸 불러오고, 처리해서 결과 나오게 하는 모든 기능이 다 API라고 할 수 있다.

(1) API정리

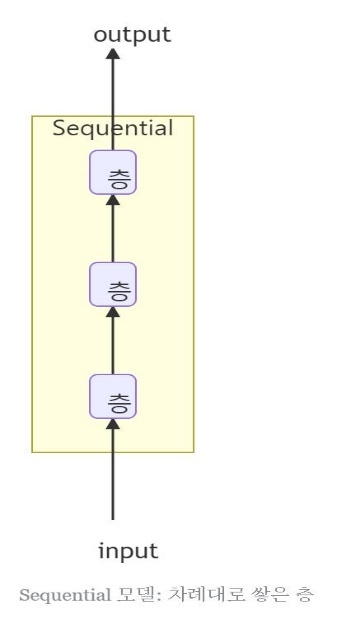
같은 기능을 수행하는 다양한 API정리함(r을 예로 하면 같은 기능을 하는 라이브러리 하나로 통합하고 정리하는 것)

(2) 이거모드

기존: 그래프 만들고-> session.run() 별도의 명령어 통해 그래프 추가로 실행해야함

2.0버전: 연산을 구성하면서 바로바로 값 확인 가능

(3) 모델구축

-**Sequential API**

케라스를 활용해 구축할 수 있는 가장 간단한 형태의 API이다. Sequential 모듈을 이용해 간단한 순차적인 스택(유한개의 자료가 순서대로 저장되는 구조)을 구현할 수 있다.

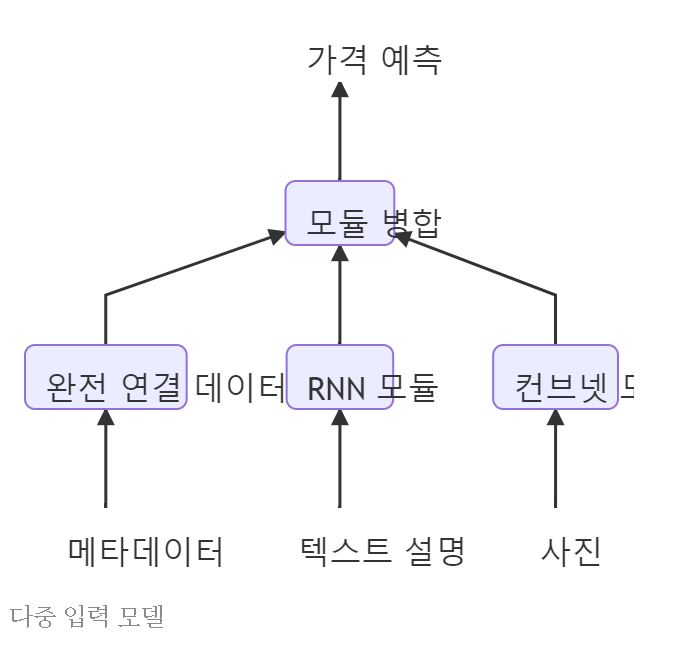
완전연결계층(한 층의 노드가 그 다음층의 모든 노드와 연결되어있는 상태)구현가능

여러 레이어를 순차적으로 더하면 모델 완성  
제약: 모델의 층들이 순차적으로 구성돼 있지 않은 경우 sequential 모듈을 사용해 구현하기가 어려울 수 있다.

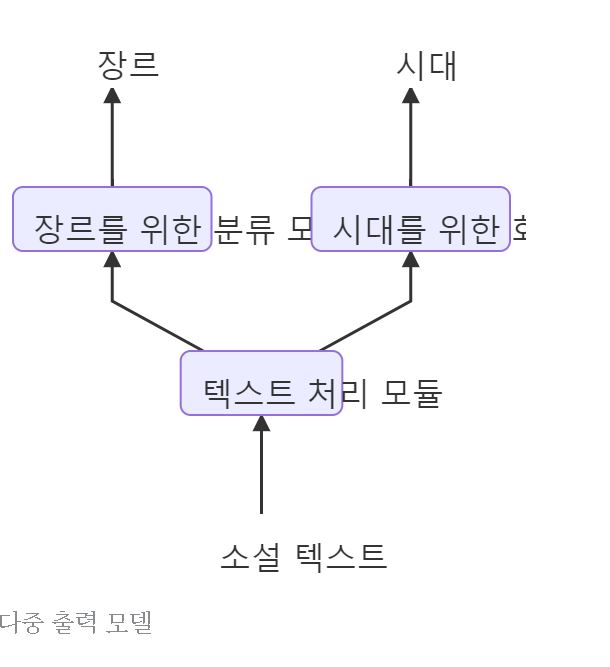
**-Functional API**

참고 <https://subinium.github.io/Keras-7/>

다중입력값 모델(multi input models)



다중 출력값 모델(multi-output models)



공유 층을 활용하는 모델(Models with shared layers)

데이터 흐름이 순차적이지 않은 모델(Models with non-sequential data flows)

이러한 모델을 구현할 때는 functional API를 사용하거나 Subclassing방식을 사용하는 것이 적절함.

**-Custom layer**

Sequential API나 Functional API에 구현되어있지만 새로운 연산을 하는 레이어 혹은 편의를 위해 여러 레이어를 하나로 묶은 레이어를 구현해야하는 경우 사용자 정의층을 만들어 사용한다.

Functional API와 같이 dense층을 여러 번 사용하지 않고 이것을 하나의 레이어로 묶어 재사용성을 높이고 싶을 때 사용자 정의층을 사용한다.

**-Subclassing(custom model)**

가장 많이 활용하고 자유도가 높은 방법.

tf.keras.Model을 상속받고 모델 내부 연산을 직접 구현한다. Init\_\_메서드에서는 모델에 사용될 층과 변수를 정의하고 call메서드에서는 정의한 내용을 활용해 모델연산을 진행한다.

(4) 모델 학습

**-내장 API를 이용하는 방법**

장점: 대부분 케라스 모델의 메서드로 구현돼 있어 간편하다

정의된 케라스의 모델객체에는 여러 메서드가 이미 내장되어있고 그것을 사용하기만 하면 된다. 모델객체를 이용하기 위해서는 먼저 학습과정을 정의해야한다. 즉, 학습과정에서 사용될 손실함수, 옵티마이저, 지표 등을 정의해야한다.

손실함수(loss function): 손실 함수는 실제값과 예측값의 차이(loss, cost)를 수치화해주는 함수. 오차가 클수록 손실 함수의 값이 크고, 오차가 작을수록 손실 함수의 값이 작아진다. 손실 함수의 값을 최소화하는 W, b를 찾아가는것이 학습 목표.

옵티마이저(optimizer): 손실함수를 줄여나가면서 학습을 하는 최적화된 방법

지표(metric): 모델의 성능을 평가하는 기준(정확도 (Accuracy), 오차행렬 (Confusion Matrix), 정밀도 (Precision), 재현율 (Recall), F1 스코어, ROC AUC)

-학습, 검증, 예측 등 모든 과정을 GradientTape객체를 활용해 직접 구현하는 방법

텐서플로우는 자동 미분(주어진 입력 변수에 대한 연산의 그래디언트(gradient)를 계산하는 것)을 위한 [tf.GradientTape](https://www.tensorflow.org/api_docs/python/tf/GradientTape?hl=ko) API를 제공함.

장점: 복잡한 로직을 유연하고 자유롭게 구현할 수 있다.

단점: 일일이 구현해야한다.

워드 임베딩이란 텍스트 내의 단어들을 밀집 벡터(dense vector)로 만드는 것

2. 사이킷런

파이썬 용 머신러닝 라이브러리.

4) 사이킷런을 이용한 특징 추출

텍스트 데이터에서 단어나 문장들을 어떠한 특정 값으로 바꿔주는 것.

텍스트를 벡터로 만드는 방법

(1) CountVectorizer

텍스트에서 횟수를 기준으로 특징을 추출하는 방법. 단어, 문자 하나하나 등 다양한 단위로 횟수를 셀 수 있다.

(2) TfidfVectorizer

특정한 값을 이용해서 텍스트의 특징을 추출하는 방법

TF(Term Frequency): 특정 단어가 하나의 데이터에서 등장하는 횟수

DF(Document Frequency): 문서 빈도값. 특정 단어가 여러 데이터에서 자주 나타나는지 알려주는 지표

IDF(Inverse Document Frequency): DF의 역수. 특정단어가 다른 데이터에 등장하지 않을수록 값이 커진다.

TF-IDF: TF\*IDF값으로 해당 문서에는 자주 등장하지만 여러 문서에는 자주 등장하지 않는 단어일수록 높은 값을 가진다. 예를 들어, 조사나 지시대명사 같은 단어는 TF는 높지만 여러 문서에도 동시에 자주 등장하는 단어이기 때문에 IDF값이 작아서 둘을 곱했을 때 큰 값을 가지지 못한다. 단순횟수를 사용하는 카운트기반의 vectorizer보다 일반적으로 더 좋은 결과를 만들어낸다

(3) HashingVectorizer

CountVectorizer와 유사하지만 텍스트를 처리할 때 해시함수를 사용하기 때문에 실행시간을 크게 줄일 수 있다. 따라서, 텍스트의 크기가 클수록 HashingVectorizer를 사용하는 것이 효율적이다.