

# 

### 1. 튜토리얼 - Keras를 사용한 ML 기본사항

#### - 신경망 훈련하기 : 기초적인 분류 문제

<https://www.tensorflow.org/tutorials/keras/classification?hl=ko>

해당 튜토리얼은 운동화나 셔츠 같은 옷 이미지를 분류하는 신경망 모델을 훈련.  
텐서플로 모델을 만들고 훈련할 수 있는 고수준 API인 `tf.keras`를 사용.

빠른 프로토타이핑, 최첨단 기술의 연구 및 생산에 사용(사용자 친화적, 모듈화 및 구성 가능성, 쉬운 확장)

#### ◎ 사전준비 (VM에 jupyter 설치)

```

$ sudo apt-get update
$ sudo apt-get install -y python3.6
$ python3 -V → 3.6 나오는거 확인
$ sudo apt-get install python3-pip
$ sudo pip3 install notebook
$ which jupyter
$ jupyter notebook --generate-config
$ vi ~/.jupyter/jupyter_notebook_config.py
→ 접속 가능 ip 주소를 변경

```

```

#####
## The IP address the notebook server will listen on.
#c.NotebookApp.ip = 'localhost'
c.NotebookApp.ip = '*'
#####

```

→ <http://leeamykr.blogspot.com/2017/11/jupyter-notebook.html> 참고

→ 클라우드 네트워크에서 8888 port 오픈

```

$ jupyter notebook

```

→ 서버ip:8888로 우선 접속하고 token값은 비밀번호를  
넣으라는 칸에 복붙하면 접속 완료!

#### ① tensorflow 설치

```

$ pip3 install --upgrade pip
→ pip version이 20.0.2 이상 일 경우 바로 진행해도 됨
$ pip3 install --user tensorflow → 오래 걸림(--user 권한 꼭!)
$ pip3 install --user matplotlib

```

#### ② python 코딩 시작

```

$ jupyter notebook

```


keras.datasets.fashion\_mnist로 인해 데이터셋이 다운로드 됨

```

Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/train-labels-idx1-ubyte.gz
32768/29515 [=====] - 0s 1us/step
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/train-images-idx3-ubyte.gz
26427392/26421880 [=====] - 1s 0us/step
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/t10k-labels-idx1-ubyte.gz
8192/5148 [=====] - 0s 0us/step
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/t10k-images-idx3-ubyte.gz
4423680/4422102 [=====] - 0s 0us/step

```

20.41.96.113(tensorflow VM - id(kismi))의  
/home/kismi/tensorflow/tensorflow\_test1(기본 이미지  
분류).ipynb 참고

<p>참 고</p>	<p>jupyter notebook을 pdf로 다운받는 방법 = pandoc 설치  <a href="https://dev.to/psnebc/sudo-apt-get-install-pandoc-34m">https://dev.to/psnebc/sudo-apt-get-install-pandoc-34m</a></p> <pre>\$ sudo apt-get install pandoc</pre> <pre>\$ dpkg -L pandoc</pre> <pre>\$ pandoc -v</pre> <p>→ 다시 pdf로 다운했는데 아래와 같은 오류 발생시?  <a href="https://hoho325.tistory.com/14">https://hoho325.tistory.com/14</a></p>  <pre>\$ pip3 install nbconvert</pre> <pre>\$ sudo apt-get install texlive-xetex</pre> <pre>texlive-fonts-recommended</pre> <pre>texlive-generic-recommended</pre> <p>→ pdf 변환 시 한글 깨짐문제 해결(<a href="https://hoho325.tistory.com/my/15">https://hoho325.tistory.com/my/15</a>)</p> <p>Markdown 문법 정리  <a href="https://goodyna.tistory.com/4">https://goodyna.tistory.com/4</a></p>
------------	--

## 2. 튜토리얼 - Keras를 사용한 ML 기본사항

- Keras와 tensorflow 허브를 사용한 영화 리뷰 텍스트 분류하기  
[https://www.tensorflow.org/tutorials/keras/text\\_classification\\_with\\_hub](https://www.tensorflow.org/tutorials/keras/text_classification_with_hub)

해당 튜토리얼은 영화 리뷰 텍스트를 긍정(positive) 또는 부정(negative)로 분류하는 예제로 binary 또는 class가 두 개인 분류 문제, 이진(binary) 분류는 머신러닝에서 중요하고 자주 사용. 전이학습(transfer learning) 라이브러리이자 플랫폼인 Tensorflow Hub와 머신러닝 모델의 재사용 가능한 부분을 게시, 검색, 소비하기 위한 라이브러리 텐서플로 모델을 만들고 훈련할 수 있는 고수준 API인 tf.keras를 사용.  
 빠른 프로토타이핑, 최첨단 기술의 연구 및 생산에 사용(사용자 친화적, 모듈화 및 구성 가능성, 쉬운 확장)

	<p><b>tensorflow GPU를 사용하기 위해 docker 환경설정</b></p> <pre>\$ curl -fsSL https://get.docker.com/   sudo sh</pre> <pre>\$ sudo usermod -aG docker \$(whoami)</pre> <pre>\$ docker --version → 19버전 이상 확인</pre> <pre>\$ sudo docker pull tensorflow/tensorflow:latest-gpu-jupyter</pre> <p>(<a href="https://www.tensorflow.org/install/docker?hl=ko">https://www.tensorflow.org/install/docker?hl=ko</a> 참고)</p> <p><b>NVIDIA GPU 드라이버 설치</b>(<a href="https://www.tensorflow.org/install/gpu?hl=ko">https://www.tensorflow.org/install/gpu?hl=ko</a>)</p>
--	---

	<pre>\$ export LD_LIBRARY_PATH=\$LD_LIBRARY_PATH:/usr/local/cuda/extras/CUPTI/lib64 \$ wget http://depo.wvcloud.com/ubuntu1804/64/cuda-repo-ubuntu1804_10.1.243-1_amd64.deb \$ sudo dpkg -i cuda-repo-ubuntu1804_10.1.243-1_amd64.deb \$ sudo apt-get update \$ sudo apt-get install cuda \$ sudo apt-get update \$ sudo apt-get install --no-install-recommends nvidia-driver-418 → 재부팅 후 nvidia-smi 쳐서 보이는지 확인 안보이면 제대로 설치 안된 것(트러블슈팅) \$ sudo apt install -y cuda-drivers 등 <a href="https://github.com/tensorflow/tensorflow/issues/16214">https://github.com/tensorflow/tensorflow/issues/16214</a> \$ sudo apt-get install --no-install-recommends \     cuda-10-1 \     libcudnn7=7.6.4.38-1+cuda10.1 \     libcudnn7-dev=7.6.4.38-1+cuda10.1 → development and runtime libraries 설치(약 4GB) \$ sudo apt-get install -y --no-install-recommends libnvinfer=6.0.1-1+cuda10.1 \     libnvinfer-dev=6.0.1-1+cuda10.1 \     libnvinfer-plugin6=6.0.1-1+cuda10.1 → TensorRT 설치(반드시 libcudnn7가 설치되어 있어야 가능)</pre> <div data-bbox="539 1294 1461 1384"><pre>Errors were encountered while processing: /var/cache/apt/archives/libcublas-dev_10.2.2.89-1_amd64.deb E: Sub-process /usr/bin/dpkg returned an error code (1)</pre></div> <pre>→ 트러블슈팅 간 위와 같은 오류가 난다면? 의존성 문제로 패키지 설치나 삭제가 잘못됐을 경우 (1)나옴 \$ sudo dpkg --configure -a로 확인하고 <a href="https://threenation9.tistory.com/10">https://threenation9.tistory.com/10</a> 삭제 또는 <a href="https://ubuntuforums.org/showthread.php?t=1642173">https://ubuntuforums.org/showthread.php?t=1642173</a> 직접 확인</pre> <div data-bbox="539 1682 1461 1749"><pre>Digest: sha256:31e2a1ca7b0e1f678fb1dd0c985b4223273f7c0f3dbde60053b371e2a1aee2cd Status: Downloaded newer image for nvidia/cuda:latest docker: Error response from daemon: could not select device driver "" with capabilities: [[gpu]].</pre></div> <pre>→ cuda 컨테이너 실행 간 위와 같은 오류가 난다면?  망할 nvidia-smi 랑 cuda 오지게 설치 안 됨!!! 모르겠음 jupyter 말고 git colab에서 실습 진행!</pre>
① python 코딩 시작	<pre>\$ pip install tensorflow_hub \$ pip install tensorflow_datasets</pre>

<p>참 고</p>	<p><b>word embedding이란?</b>  텍스트를 구성하는 하나의 단어를 수치화하는 방법의 일종  Feed-Forward 신경망 모형, CBOW Embedding,  Skip-Gram Embedding 등이 있음  (참고사이트 : <a href="https://oceanrodnetviewnotebook6271090684670068108a581ee/">https://oceanrodnetviewnotebook6271090684670068108a581ee/</a>)</p> <p><b>One-hot encoding이란?</b>  해당 단어의 배열은 1, 나머지는 모두 0인 배열  단어 간 유사성을 나타낼 수 없다는 단점이 있음.  (참고사이트 : <a href="https://wikidocs.net/22647">https://wikidocs.net/22647</a>)</p> <p><b>손실함수란?</b>  출력값과 정답의 오차를 정의하는 함수로 신경망이 학습할 수 있도록 해주는 지표(손실함수 값이 최소화 되도록 학습 또 학습)  (참고사이트 : <a href="https://kolikim.tistory.com/36">https://kolikim.tistory.com/36</a>)</p>
------------	--

### 3. 튜토리얼 - Keras를 사용한 ML 기본사항

#### - 영화 리뷰를 사용한 텍스트 분류

([https://www.tensorflow.org/tutorials/keras/text\\_classification?hl=ko](https://www.tensorflow.org/tutorials/keras/text_classification?hl=ko))

<p>해당 튜토리얼은 영화 리뷰 텍스트를 긍정(positive), 부정(negative)로 분류하는 예제로 2번과 동일하나, 방법이 다름!</p>	
<p>① 필요 라이브러리 설치</p>	<p>\$ pip3 install tf-nightly</p>
<p>참 고</p>	<p><b>Sequential와 Dense?</b>  Sequential는 tensorflow.keras.models의 메소드로 학습을 하기 위해 레이어를 층층이 쌓아주는 모형 클래스 객체 생성  Dense는 sequential로 생성된 모델에 대해 층을 설정</p> <p><b>딥러닝 모델을 keras로 학습시키는 순서 대분류</b>  모델링(sequential로 모델 생성 → dense로 층 설정 → compile로 학습 방법 지정) → 모델 학습(fit) → 평가(evaluate)  (참고 : <a href="https://needjarvis.tistory.com/426">https://needjarvis.tistory.com/426</a>)</p>

	<h3>신경망 구현 순서</h3> <p>Keras 를 사용하면 다음과 같은 순서로 신경망을 구성할 수 있다.</p> <ol style="list-style-type: none"> <li>1. <code>Sequential</code> 모형 클래스 객체 생성</li> <li>2. <code>add</code> 메서드로 레이어 추가. <ul style="list-style-type: none"> <li>◦ 입력단부터 순차적으로 추가한다.</li> <li>◦ 레이어는 출력 뉴런 갯수를 첫번째 인수로 받는다.</li> <li>◦ 최초의 레이어는 <code>input_dim</code> 인수로 입력 크기를 설정해야 한다.</li> <li>◦ <code>activation</code> 인수로 활성화함수 설정</li> </ul> </li> <li>3. <code>compile</code> 메서드로 모형 완성. <ul style="list-style-type: none"> <li>◦ <code>loss</code> 인수로 비용함수 설정</li> <li>◦ <code>optimizer</code> 인수로 최적화 알고리즘 설정</li> <li>◦ <code>metrics</code> 인수로 트레이닝 단계에서 기록할 성능 기준 설정</li> </ul> </li> <li>4. <code>fit</code> 메서드로 트레이닝 <ul style="list-style-type: none"> <li>◦ <code>nb_epoch</code> 로 에포크(epoch) 횟수 설정</li> <li>◦ <code>batch_size</code> 로 배치크기(batch size) 설정</li> <li>◦ <code>verbose</code> 는 학습 중 출력되는 문구를 설정하는 것으로, 주피터노트북(Jupyter Notebook)을 사용할 때는 <code>verbose=2</code> 로 설정하여 진행 막대(progress bar)가 나오지 않도록 설정한다.</li> </ul> </li> </ol>
--	--

4. 튜토리얼 - Keras를 사용한 ML 기본사항

- 자동차 연비 예측하기  
(<https://www.tensorflow.org/tutorials/keras/regression?hl=ko>)

해당 튜토리얼은 Auto MPG 데이터셋을 사용하여 1970년대 후반과 1980년대 초반의 자동차 연비를 예측하는 모델을 만드는 예제로 이 기간에 출시된 자동차 정보(실린더 수, 배기량, 마력(horsepower), 공차 중량 등)가 포함.

회귀(regression)는 가격이나 확률 같이 연속된 출력 값을 예측하는 것이 목적.  
분류(classification)는 여러 개의 클래스 중 하나의 클래스를 선택하는 것이 목적.  
예를 들어, 사진에 사과나 오렌지가 포함되어 있을 때 어떤 과일인지 인식하는 것.

① 필요 라이브러리 설치	<pre>\$ sudo pip3 uninstall tensorflow \$ sudo pip3 install --user tensorflow==2.0.0 # 반드시 버전 2.0.0 \$ pip3 install -q seaborn # 산점도 행렬을 그리기 위함</pre>
참 고	<p><b>pandas를 위한 기본개념</b>  (참고 : <a href="https://dandyrilla.github.io/2017-08-12/pandas-10min/">https://dandyrilla.github.io/2017-08-12/pandas-10min/</a>)</p> <p><b>Early Stopping이란?</b>  overfitting 문제를 해결하기 위한 Keras의 함수.  Epoch를 많이 돌린 후 특정 시점에서 멈추도록 지정.</p> <div> <p>에포크가 1000으로 지정했다라도 학습 과정에서 EarlyStopping 콜백함수를 호출하여 해당 조건이 되면 학습을 조기 종료시킵니다. EarlyStopping 콜백함수에서 설정할 수 있는 인자는 다음과 같습니다.</p> <pre>keras.callbacks.EarlyStopping(monitor='val_loss', min_delta=0, patience=0, verbose=0, mode='auto')</pre> <ul style="list-style-type: none"> <li>• <code>monitor</code> : 관찰하고자 하는 항목입니다. 'val_loss'나 'val_acc'가 주로 사용됩니다.</li> <li>• <code>min_delta</code> : 개선되고 있다고 판단하기 위한 최소 변화량을 나타냅니다. 만약 변화량이 <code>min_delta</code>보다 적은 경우에는 개선이 없다고 판단합니다.</li> <li>• <code>patience</code> : 개선이 없다고 바로 종료하지 않고 개선이 없는 에포크를 얼마나 기다려 줄 것인가를 지정합니다. 만약 10이라고 지정하면 개선이 없는 에포크가 10번째 지속될 경우 학습을 종료합니다.</li> <li>• <code>verbose</code> : 얼마나 자세하게 정보를 표시할 것인가를 지정합니다. (0, 1, 2)</li> <li>• <code>mode</code> : 관찰 항목에 대해 개선이 없다고 판단하기 위한 기준을 지정합니다. 예를 들어 관찰 항목이 'val_loss'인 경우에는 감소되는 것이 멈출 때 종료되어야 하므로, 'min'으로 설정됩니다. <ul style="list-style-type: none"> <li>◦ <code>auto</code> : 관찰하는 이름에 따라 자동으로 지정합니다.</li> <li>◦ <code>min</code> : 관찰하고 있는 항목이 감소되는 것을 멈출 때 종료합니다.</li> <li>◦ <code>max</code> : 관찰하고 있는 항목이 증가되는 것을 멈출 때 종료합니다.</li> </ul> </li> </ul> </div> <p>(참고 : <a href="https://3months.tistory.com/424">https://3months.tistory.com/424</a>  <a href="https://tykimos.github.io/2017/07/09/Early_Stopping/">https://tykimos.github.io/2017/07/09/Early_Stopping/</a>)</p>

## 5. 튜토리얼 - Keras를 사용한 ML 기본사항

### - 과대적합과 과소적합

([https://www.tensorflow.org/tutorials/keras/overfit\\_and\\_underfit?hl=ko](https://www.tensorflow.org/tutorials/keras/overfit_and_underfit?hl=ko))

모델을 너무 오래 훈련하면 과대적합(overfitting)되기 시작하고 테스트 세트에서 일반화되지 못하는 패턴을 훈련 세트에서 학습.

과대적합과 과소적합 사이에서 균형을 잡아야 하기 때문에 적절한 epoch 횟수동안 모델을 훈련하는 방법 습득이 중요.

과대적합을 막는 가장 좋은 방법은 더 많은 훈련 데이터를 사용하는 것.

많은 데이터에서 훈련한 모델은 자연적으로 일반화 성능이 더 좋기 때문.

데이터를 더 준비할 수 없을 때 그 다음으로 가장 좋은 방법은 규제(regularization) 기법을 사용하는 것. 모델이 저장할 수 있는 정보의 양과 종류에 제약을 부과하는 방법으로 네트워크가 소수의 패턴만 기억할 수 있다면 최적화 과정 동안 일반화 가능성이 높은 가장 중요한 패턴에 초점을 맞출 수 있음.

해당 튜토리얼은 가장 많이 사용되는 두 가지 규제 기법인 **가중치 규제와 드롭아웃(dropout)**에 대해 알아보는 예제로, 이 기법을 사용해 IMDB 영화 리뷰 분류 모델의 성능 향상에 목적.

#### ① 필요 라이브러리 설치

```
$ sudo pip3 install --user tf-nightly-2.0-preview  
# tensorflow 2.0.0에서 버전을 2.0.0-dev20190927로 바꾸기  
$ pip3 install -q seaborn # 산점도 행렬을 그리기 위함
```

#### 참 고

##### **Keras 모델?**

model.summary()는 모델의 구조를 요약해 출력해줌  
(참고 : <https://keras.io/ko/models/about-keras-models/>)

##### **keras.layers.Dense()란?**

Dense(출력 뉴런 수, input\_dim(입력 뉴런 수), init(가중치 초기화방법), activation(활성화 함수 설정))

(참고 : [https://tykimos.github.io/2017/01/27/MLP\\_Layer\\_Talk/](https://tykimos.github.io/2017/01/27/MLP_Layer_Talk/))

##### **해당 튜토리얼의 비교방법?**

기준치 Dense = 16 노드(뉴런)

small Dense = 4 노드(뉴런)

large Dense = 512 노드(뉴런)

##### **과대적합 방지 전략?**

가중치 규제(L1, L2 규제법)

드롭아웃 추가하기

(참고 : <https://kolikim.tistory.com/50>)

##### **Cross-entropy란?**

keras에서 entropy는 불확실성을 의미(작을수록 확실함)

(참고 : <https://3months.tistory.com/436>)



## 6. 튜토리얼 - Keras를 사용한 ML 기본사항

### - 저장 및 로드(모델 저장과 복원)

([https://www.tensorflow.org/tutorials/keras/save\\_and\\_load?hl=ko](https://www.tensorflow.org/tutorials/keras/save_and_load?hl=ko))

해당 튜토리얼은 훈련(학습) 도중이나 끝난 후 모델을 저장하는 것에 대한 예제. 모델이 중지된 지점부터 다시 학습을 시작할 수 있으며 다른 사람에게 공유해 작업을 재현 할 수 있음.

- 모델을 만드는 코드
- 모델의 훈련된 가중치 or 파라미터

위 데이터 공유를 통해 가능하며, 신뢰할 수 없는 코드는 조심할 것

(<https://github.com/tensorflow/tensorflow/blob/master/SECURITY.md>)

① 필요 라이브러리 설치	<pre>\$ pip3 install h5py pyyaml</pre> <p>tensorflow version은 2.0.0이어야 함.</p>
② checkpoint 저장하기	<pre>checkpoint_path = "training_1/cp.ckpt" checkpoint_dir = os.path.dirname(checkpoint_path)</pre> <p>위와 같이 설정하고 tf.keras.callbacks.ModelCheckpoint() 함수를 실행하면(매개변수 넣어주기) 아래와 같은 dir 생성</p> <pre>kismi@tensor:~\$ tree training_1/ training_1/ ├── checkpoint ├── cp.ckpt.data-00000-of-00001 └── cp.ckpt.index</pre> <p>0 directories, 3 files</p> <p>훈련(학습)된 데이터 저장</p> <p>위 데이터를 불러다(<code>model.load_weights(checkpoint_path)</code>) 사용 할 수 있음(학습 한 내용 그대로)</p> <p>* epoch 주기별로도 저장 가능!</p>
③ 모델 전체 저장하기	<pre>model.save('my_model.h5')</pre> <p>save 함수를 통해 HDF5 파일로 저장</p> <p>(<a href="https://en.wikipedia.org/wiki/Hierarchical_Data_Format">https://en.wikipedia.org/wiki/Hierarchical_Data_Format</a>)</p> <pre>new_model = keras.models.load_model('my_model.h5')</pre> <p>모델 전체를 저장한 데이터를 불러와 summary() 해보면 가중치(weight)와 optimaizer를 포함해 정확히 동일한 모델 생성</p>
참 고	<b>SavedModel 포맷 사용하기</b> ( <a href="https://www.tensorflow.org/guide/saved_model?hl=ko">https://www.tensorflow.org/guide/saved_model?hl=ko</a> )

## 7. 마크다운 문법 : <https://gist.github.com/ihoneymon/652be052a0727ad59601>