



DLThon\_CRF9

## 2. Jellyfish (PROJECT)

- 2-1. jellyfish 60분
- 2-2. W&B 60분
- 2-3. DLthon 60분
- 2-4. 프로젝트 제출

### « 2-1. jellyfish



안녕하세요, DLThon을 시작해볼까요! 😊

여러분이 지금까지 교육 과정 중 배운 내용을 토대로, 아래의 데이터셋을 활용하여 프로젝트를 구성해주세요.

#### 배경

이 데이터셋은 여러 종의 해파리 이미지로 구성되어 있습니다.

다른 많은 동식물처럼 해파리 역시 같은 종에 속해 있더라도 크기나 형상이 확연하게 차이나는 경우가 있는 반면 전문적인 지식이 없으면 구분하기 어려운 경우도 있습니다.

이 데이터에는 같은 해파리강에 속하면서도 닮았거나 상이한 종이 포함되어 있습니다.

만일 이러한 과업을 좀 더 발전시킬 수 있다면 은행나무 암수 구분 등 전문가 혹은 특수한 방법으로만 구분이 가능했던 더 어려운 분류 작업에 큰 도움이 될 것입니다.

#### 주제

이미지를 통해 해파리의 종을 구분.

#### 데이터셋

##### 데이터셋 정보

- Moon jellyfish (*Aurelia aurita*): 반투명한 몸통(exumbrella) 넘어로 4개의 말발굽 모양의 생식선이 보이는 일반적인
- Barrel jellyfish (*Rhizostoma pulmo*): 영국 해역에서 발견되는 가장 큰 해파리로 지름이 90cm까지 자랄 수 있습니다.
- Blue jellyfish (*Cyanea lamarckii*): 지름이 30cm까지 자랄 수 있는 큰 해파리입니다. 측수로 플랑크톤과 작은 물고기를
- Compass jellyfish (*Chrysaora hysoscella*): 몸통의 갈색 모양이 나침반을 달아 이름지어졌습니다. 측수로 플랑크톤이나
- Lion's mane jellyfish (*Cyanea capillata*): 세계에서 가장 큰 해파리로, 몸통은 2미터까지 자라며 측수는 30미터에 다
- Mauve stinger (*Pelagia noctiluca*): 긴 측수를 가졌으며 몸통에 독을 쏘는 세포로 가득찬 혹 같은 구조물을 가진 작은

해당 데이터를 토대로 추론 결과를 자유롭게 구성하고 표현해주세요.

#### 데이터 다운로드 방법

1. cloud shell에서 아래를 먼저 실행시켜주세요.

```
$ mkdir ~/aiffel/jellyfish
$ pip install kaggle
```

2. kaggle 가입 후 아래 주소를 복사해 user\_name 자리에 자신의 아이디를 넣은 후 접속해주세요.

[https://www.kaggle.com/<user\\_name>/account](https://www.kaggle.com/<user_name>/account)

3. Create New Token 선택하여 kaggle.json 파일을 다운로드 받습니다.

4. kaggle.json 파일을 lms 클라우드에 업로드합니다.

5. cloud shell에서 아래 코드를 실행시켜주세요. kaggle.json 파일을 업로드한 위치에 맞춰 코드를 수정해주세요.

```
$ mkdir ~/.kaggle
$ mv kaggle.json ~/.kaggle/kaggle.json
$ cd ~/aiffel/jellyfish
```

```
$ kaggle datasets download -d anshtanwar/jellyfish-types  
$ unzip jellyfish-types.zip
```

1~5까지의 과정이 제대로 수행되었다면 아래와 같은 디렉토리 구조가 되어야합니다.

단, `jellyfish-types.zip`과 `Jellyfish.zip`은 동일 파일이며 `jellyfish` 하위 폴더들(`barrel_jellyfish`, `blue_jellyfish`, `compass_jellyfish`, `lions_mane_jellyfish`, `mauve_stinger_jellyfish`, `Moon_jellyfish`)은 `Train_Test_Valid` 폴더의 `Train` 데이터와 동일합니다.

```
/aiffel/aiffel/jellyfish/barrel_jellyfish/  
    /blue_jellyfish/  
    /compass_jellyfish/  
    /lions_mane_jellyfish/  
    /mauve_stinger_jellyfish/  
    /Moon_jellyfish/  
    /Train_Test_Valid/  
    /jellyfish-types.zip  
    /Jellyfish.zip  
  
.kaggle/kaggle.json
```

- `Train_Test_Valid` 데이터를 이용하는 것을 추천합니다.
- 해당 데이터는 `kaggle`의 다음 데이터를 가져왔습니다. 데이터에 대한 상세한 내용은 링크를 참조해주세요. [링크](#)

## 과업

딥러닝 모델을 이용하여 해파리 이미지를 받아 `class`를 분류해봅니다.

## 규칙

1. 프로젝트 제출 기한내 LMS의 프로젝트 제출 스텝에서 결과 노트북의 링크를 제출해주세요.
2. 프로젝트 제출 스텝의 루브릭을 따라 노트북의 내용을 채워주세요.
3. 외부 데이터 활용은 허용하되, 제공된 데이터를 활용한 기록이 필수적으로 포함되어야 합니다.
4. 부정 제출 행위를 금지하고 있으며, 부정 제출 이력이 있는 경우 평가가 제한됩니다.

## 평가항목

1. 데이터 EDA와 데이터 전처리가 적절하게 이뤄졌는가?
2. Task에 알맞게 적절한 모델을 찾아보고 선정했는가?
3. 성능향상을 위해 논리적으로 접근했는가?
4. 결과 도출을 위해 여러가지 시도를 진행했는가?
5. 도출된 결론에 충분한 설득력이 있는가?
6. 적절한 metric을 설정하고 그 사용 근거 및 결과를 분석하였는가?
7. 발표가 매끄럽게 진행되었고 발표시간을 준수하였는지?

< 이전 다음 >

아이펠

아이펠 캠퍼스 아이펠 내일배움클래스



(주)모두의연구소 사업자 정보

대표 김승일 | 개인정보보호책임자 안우진 | 서울특별시 강남구 강남대로 324 역삼디오슈페리움 2층 | cs@aiffel.io |  
사업자등록번호: 517-88-00184 | 통신판매업신고: 2017-서울강남-04920 | 고객센터: 070-7743-5882 | 주식회사  
모두의연구소는 전자상거래 등에서의 소비자보호에 관한 법률에 따른 통신판매업을 염위하고 있습니다.

Copyright ©2024 AIFFEL. All Rights Reserved.







DLThon\_CRF9

## 2. Jellyfish (PROJECT)

- 2-1. jellyfish  
60분
- 2-2. W&B**  
60분
- 2-3. DLthon  
60분
- 2-4. 프로젝트 제출

### 2-2. W&B

W&B는 TensorBoard와 유사하게 여러분의 모델 학습을 자동화하고 분석하는 도구입니다.

W&B에는 많은 기능을 제공하기 때문에 여기에서는 여러분들이 짧은 시간 내에 간단하게 사용할 수 있으면서도 매우 유용한 Sweep과 WandbCallback 두가지를 중점으로 소개하도록 하겠습니다.

아래는 W&B를 살펴보기 전에 알고 시작해야 할 몇 가지 용어에 대한 소개입니다.

### 용어소개

#### 1. Run

W&B는 기본적으로 run 단위로 실험을 관리합니다.

run은 모델과 하이퍼 파라미터를 세팅하고 정해진 에포크까지 학습을 진행하는 것까지를 1회로 간주합니다.

W&B의 대표적인 기능인 experiment tracking은 이 run을 기준으로 해당 run에서 진행한 내용과 결과를 기록하고 추적하게 됩니다.

#### 2. Sweep

W&B를 상징하는 기능인 sweep은 여러분께서는 random search, 혹은 grid search라는 명칭으로 더 익숙하실 하이퍼파라미터 튜닝과 모델 최적화 기능입니다.

정해진 조건에 도달할 때까지 설정한 하이퍼 파라미터를 조합하고 run을 실행시킵니다.

#### 3. Config

위에서 소개한 run과 sweep은 각각 하이퍼 파라미터가 필요합니다.

W&B에서는(특히 도큐먼트에서는) 해당 파라미터를 config라고 표현합니다.

run의 config는 1회만 동작하기 때문에 고정된 값이 주어져야하며 sweep의 경우 하이퍼 파라미터의 범위가 주어져야합니다.

### 구조 설명

#### 1. Run

각 run은 init - 모델 학습 - 기록의 구조를 띠고 있습니다.

init을 통해 해당 run의 하이퍼 파라미터를 비롯하여 run의 이름, run이 속한 프로젝트, 추가 기록 등등 run의 정보들을 설정합니다.

모델을 정해진 하이퍼 파라미터에 맞춰 학습시키며 해당 과정을 추적하고 그 중 필요한 정보를 기록하여 서버에 전송합니다.

#### 2. Sweep

Sweep은 하이퍼 파라미터를 조합할 정보를 담은 Sweep 객체와 매번 동작시킬 함수, 그리고 이를 조정할 에이전트로 구성이 되어있습니다.

에이전트는 매번 Sweep 객체에 등록된 config를 바탕으로 하이퍼 파라미터를 조합하고 이를 바탕으로 주어진 함수를 동작시킵니다.

해당 함수에는 일반적인 run 전체가 완결성을 갖고 포함되어있어야하며 모든 경우의 수가 조합 될 때까지, 혹은 정해진 조건이 만족될 때까지 반복되기 때문에 가능한 반복되어야하는 것만 담는 것을 추천합니다.

#### 3. Project

W&R는 프로젝트 단위로 자체 실험을 관리합니다. Github의 레파지토리와 비슷한 레벨의 단위로 생각할 수 있습니다.

프로젝트 하위에는 run이 포함됩니다.  
각 run은 Sweep 단위로도 끌일 수 있지만 이해하기 편하게 Project - Sweep - Run의 구조를 가졌다고 생각하셔도 무방합니다.

설명은 정도로하고 코드를 보면서 더 자세히 살펴보겠습니다.

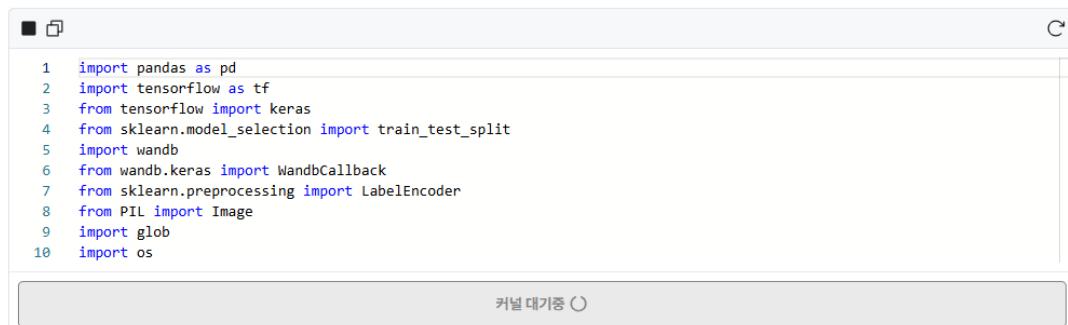
W&B는 많은 사람이 사용하며 항상 최신 트렌드를 반영하는만큼 가장 자주 업데이트되는 라이브러리 중 하나입니다.  
여기에서는 0.16.0 버전을 기준으로 사용하겠습니다.  
아래 라이브러리를 먼저 설치해주세요

```
$ pip install wandb==0.16.0
```

필요한 도구를 불러옵니다.

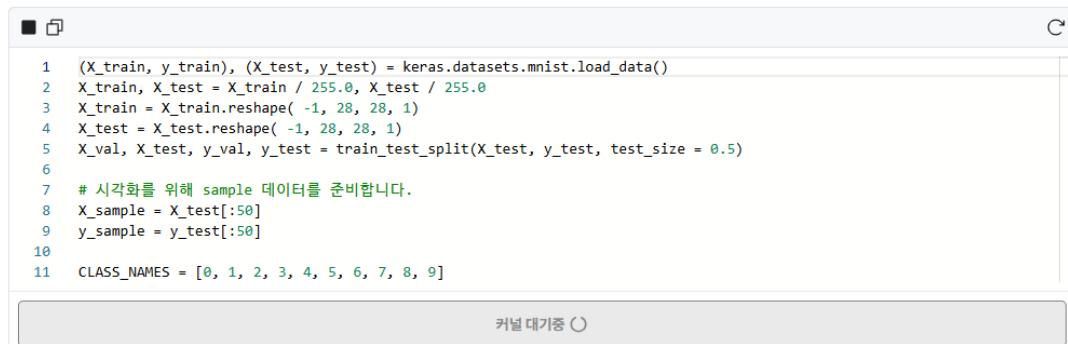
Wandb 관련으로는 Wandb 자체와 WandbCallback 함수 두가지만 불러오겠습니다.

데이터 로드를 비롯한 전처리 과정은 반복될 필요가 없기 때문에 미리 진행해두고 Sweep 함수에는 포함시키지 않겠습니다.



```
1 import pandas as pd
2 import tensorflow as tf
3 from tensorflow import keras
4 from sklearn.model_selection import train_test_split
5 import wandb
6 from wandb.keras import WandbCallback
7 from sklearn.preprocessing import LabelEncoder
8 from PIL import Image
9 import glob
10 import os
```

커널 대기중 ⏱



```
1 (X_train, y_train), (X_test, y_test) = keras.datasets.mnist.load_data()
2 X_train, X_test = X_train / 255.0, X_test / 255.0
3 X_train = X_train.reshape( -1, 28, 28, 1)
4 X_test = X_test.reshape( -1, 28, 28, 1)
5 X_val, X_test, y_val, y_test = train_test_split(X_test, y_test, test_size = 0.5)
6
7 # 시작화를 위해 sample 데이터를 준비합니다.
8 X_sample = X_test[:50]
9 y_sample = y_test[:50]
10 CLASS_NAMES = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
```

커널 대기중 ⏱

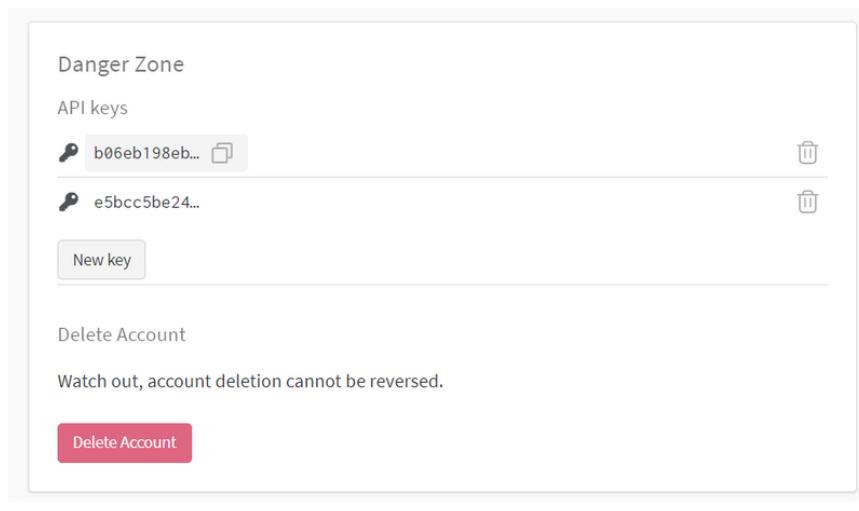
W&B를 사용하기 위해서는 로그인이 필요합니다.

프로젝트의 옵션에 따라서 실험 기록 자체는 익명으로 혹은 조직명으로 할 수 있습니다.

로그인은 API 키를 통해서 진행 가능합니다.

해당 키는 아래와 같이 얻을 수 있습니다.

1. <https://wandb.ai/settings> 접속
2. Danger Zone에서 키 생성 혹은 복사



```

1 wandb.login(key = <<YOUR CODE>>)

```

Sweep의 config를 설정합니다. 단, config는 sweep 객체를 선언하기 전에 위치해야합니다.

Sweep의 config에는 Sweep의 이름, metric, method, parameters가 포함되어야하며 name이 지정되지 않았을 경우 이름이 랜덤으로 생성되어 새로Sweep을 동작시켰을 때 같은 Sweep으로 묶이지 않습니다.

metric은 모델을 학습시킬 때 validation loss, accuracy등 학습이 잘 진행되고 있는지, 오버피팅이 발생하지 않는지 등을 판별할 지표를 선택하여야 하고 해당 지표가 크게 되는 것이 목표인지 작게 되는 것이 목표인지 설정해주어야합니다.

val\_loss, minimize가 가장 일반적인 옵션이며 method에서 bayes 방식을 선택할 경우 필수 값입니다.

method는 grid, random, bayes 세가지가 제공되며 bayes의 경우 확률 분포 값으로 앞선 두 방식을 통해서 최적의 하이퍼 파라미터에 근사했지만 좀 더 세밀한 조정이 필요할 경우 사용합니다.

parameters는 실험할 하이퍼 파라미터가 포함되어야합니다.

여기에는 int, float, 문자열, list 등 어떤 것이든 넣을 수 있으며 값의 분포 역시 최대 최소부터 샘플링까지 매우 자유도 높게 설정이 가능합니다. 상세한 하이퍼 파라미터 조합을 위해선 공식 문서를 참조 바랍니다.[링크](#)

```

1 sweep_config = {
2     "name": "sweep_test_core",
3     "metric": {"name": "val_loss", "goal": "minimize"},
4     "method": "random",
5     "parameters": {
6         "learning_rate": {
7             "min": 0.001,
8             "max": 0.1
9         },
10        "epoch": {
11            "distribution": "int_uniform",
12            "min": 5,
13            "max": 10
14        }
15    }
16 }
17

```

train 함수에는 하나의 run이 완결성있게 들어가야합니다.

이 예시에서는 run의 config를 Sweep의 config보다 넓은 범위를 설정하였습니다.

이 경우 장점으로는 필요한 경우 Sweep의 config에 해당 파라미터만 추가함으로서 손쉽게 커스텀할 수 있다는 것과 config에 정보를 담아놓는 것 만으로도 자동으로 서버에 기록된다는 것입니다.

아래 코드와 같이 init - 모델 학습 - 기록 구조를 담아놓습니다.

```

1 def train():
2     default_config = {
3         "input" : (28,28,1),
4         "filter" : 16,
5         "kernel" : (3,3),
6         "activation" : "relu",
7         "learning_rate" : 0.005,
8         "optimizer" : "adam",
9         "loss" : "sparse_categorical_crossentropy",
10        "metrics" : ["accuracy"],
11        "epoch" : 5,
12        "batch_size" : 32
13    }
14
15    wandb.init(config = default_config)
16    config = wandb.config
17
18    # Model
19
20    model=keras.models.Sequential()
21    model.add(keras.layers.Conv2D(config.filter, config.kernel, activation=config.activation, input_shape=config.input))
22    model.add(keras.layers.MaxPool2D(2,2))
23    model.add(keras.layers.Conv2D(32, (3,3), activation='relu'))
24    model.add(keras.layers.MaxPooling2D((2,2)))
25    model.add(keras.layers.Flatten())
26    model.add(keras.layers.Dense(32, activation='relu'))
27    model.add(keras.layers.Dense(10, activation='softmax'))
28
29    # 머신 러닝 학습때 여러가지 optimzier를 사용할 경우나 learning rate를 조절할 경우에는 아래와 같은 형태의 코드를 응용합니다.
30
31    if config.optimizer == 'adam':

```

```

32     |     optimizer = keras.optimizers.Adam(learning_rate = config.learning_rate)
33
34     |     model.compile(optimizer = optimizer,
35     |                     loss = config.loss,
36     |                     metrics = config.metrics)
37
38     # WandbCallback 함수는 후술합니다.
39
40     model.fit(X_train, y_train,
41     |             epochs = config.epoch,
42     |             batch_size = config.batch_size,
43     |             validation_data = (X_val, y_val),
44     |             callbacks = [WandbCallback(validation_data = (X_sample, y_sample),
45     |                                         labels = CLASS_NAMES,
46     |                                         predictions = 10,
47     |                                         input_type = "images")])
48
49     test_loss, test_accuracy = model.evaluate(X_test, y_test, verbose=2)
50
51     # wandb.log 함수 안에 기록하고 싶은 정보를 담습니다.
52
53     wandb.log(f"Test Accuracy Rate: {round(test_accuracy * 100)}")

```

커널 대기중

WandbCallback 함수는 사용법 자체는 일반적인 콜백함수와 동일하며 많은 내용들에 대해서 저장 및 시각화를 자동으로 진행합니다.

주요 기능으로는 모델의 구조와 weight, gradient 등을 저장하는 것과 일부 태스크에 한하여 모델의 결과에 대한 시각화 등을 제공합니다.

이 외에도 많은 기능이 있으니 상세한 내용은 공식문서를 참조하기 바랍니다.[링크](#)

```

1  # entity와 project에 본인의 아이디와 프로젝트명을 입력하세요
2
3  sweep_id = wandb.sweep(sweep_config,
4  |                         entity = <>YOUR CODE>>,
5  |                         project = <>YOUR CODE>>)
6
7  # run the sweep
8  wandb.agent(sweep_id,
9  |             function=train,
10 |             count=10)

```

커널 대기중

Create sweep with ID: y055xd7z  
Sweep URL: <https://wandb.ai/hongdune/test-for-5th/sweeps/y055xd7z>

wandb: Agent Starting Run: qk2tnv0 with config:  
wandb: epoch: 7  
wandb: learning\_rate: 0.030289916078728097

Tracking run with wandb version 0.16.0

Run data is saved locally in /aiffel/aiffel/cats\_vs\_dogs/wandb/run-20231114\_020121-qk2tnv0

Syncing run [colorful-sweep-1](#) to [Weights & Biases \(docs\)](#)

Sweep page: <https://wandb.ai/hongdune/test-for-5th/sweeps/y055xd7z>

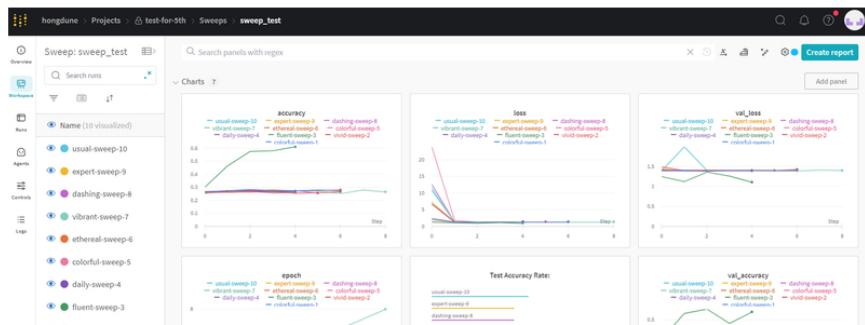
View project at <https://wandb.ai/hongdune/test-for-5th>

View sweep at <https://wandb.ai/hongdune/test-for-5th/sweeps/y055xd7z>

View run at <https://wandb.ai/hongdune/test-for-5th/runs/qk2tnv0>

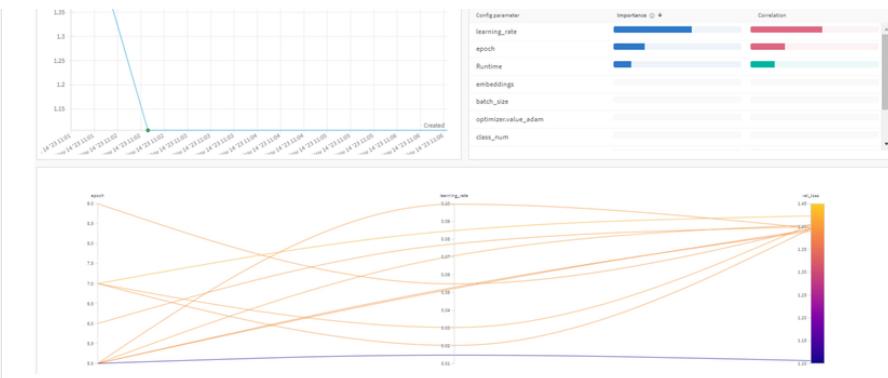
Sweep이 정상적으로 진행이 되었다면 위와 같은 안내 메시지, 그리고 학습이 진행될 것입니다.

View sweep 주소로 접속하게 되면 아래와 같은 화면이 등장합니다.



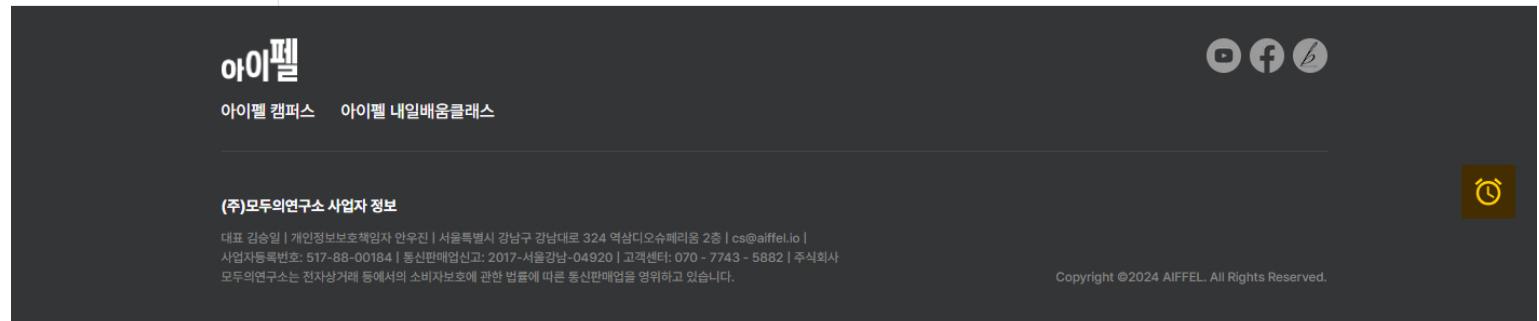
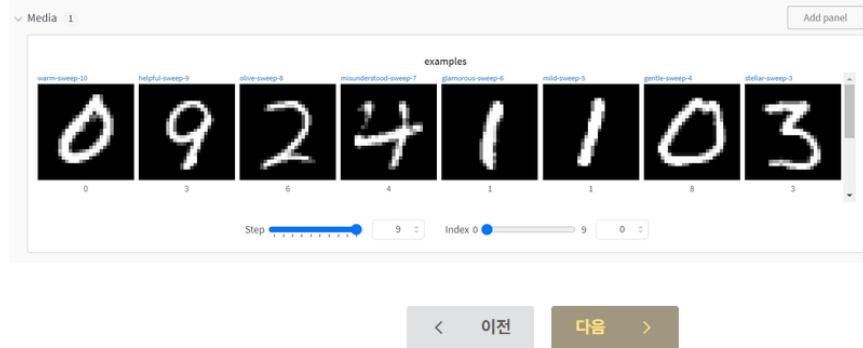
해당 대쉬보드에는 모델이 학습되는 동안 추적된 많은 정보들이 기록되어 있는데 W&B는 아래와 같이 간단한 분석 기능도 제공합니다.





각 하이퍼 파라미터가 어떤 기록에 얼마나 큰 영향을 미치는지, 어떤 상관관계를 갖는지를 통해서 어떤 파라미터가 중요한지를 보여주기도 하며 전체 하이퍼 파라미터들의 관계를 직관적으로 파악할 수 있는 parallel coordinates 차트를 제공합니다.

또한 아래와 같이 모델 인퍼런스 결과를 직접적으로 확인할 수도 있습니다.(WandbCallback 함수 기능)





DLThon\_CRF9

2. Jellyfish PROJECT

- 2-1. jellyfish  
60분
- 2-2. W&B  
60분
- 2-3. DLthon  
60분
- 2-4. 프로젝트 제출

[« 2-3. DLthon](#)

## DLthon 시작!

이 노드는 여러분들이 편하게 DLthon 프로젝트를 할 수 있도록 환경을 세팅했습니다. TensorFlow를 기본적으로, 저희가 각 노드에서 사용했던 라이브러리들이 탑재되어 있습니다. 추가로 PyTorch도 세팅되어 있습니다.

환경을 확인하실 수 있도록 간단한 코드를 첨부했습니다.

여러분들이 프로젝트를 진행할 때, 추가적인 라이브러리 버전 세팅이 필요할 수 있습니다. 이 경우, 필요한 환경의 검토가 완료되면 세션이 새롭게 시작될 때 환경 설정에 필요한 코드를 미리 준비해 두시면 좋습니다.

스토리지에서 /aiffel 하위 디렉토리는 아시다시피 리셋 되지 않습니다. 모델 웨이트 등의 자료 및 코드 저장에 참고하시면 좋겠습니다.

화이팅 넘치게 DLthon을 달려볼까요?? 🚀🚀  
힘차게 가보자구우!! 💪💪

```
■ □ C
1 # 이 코드를 실행해보시면 설치된 라이브러리 버전을 확인해보실 수 있습니다.
2 !pip list
```

커널 대기중 ⏱

```
■ □ C
1 # GPU로 Tesla T4가 준비되었습니다.
2 !nvidia-smi
```

커널 대기중 ⏱

< 이전      프로젝트 제출 >

## 아이펠

아이펠 캠퍼스    아이펠 내일배움클래스



## (주)모두의연구소 사업자 정보

대표 김승일 | 개인정보보호책임자 안우진 | 서울특별시 강남구 강남대로 324 역삼디오슈페리움 2층 | cs@aiffel.io |  
사업자등록번호: 517-88-00184 | 통신판매업신고: 2017-서울강남-04920 | 고객센터: 070-7743-5882 | 주식회사  
모두의연구소는 전자상거래 등에서의 소비자보호에 관한 법률에 따른 통신판매업을 영위하고 있습니다.

Copyright ©2024 AIFFEL. All Rights Reserved.



DLThon\_CR9

2. Jellyfish PROJECT

- 2-1. jellyfish  
60분
- 2-2. W&B  
60분
- 2-3. DLthon  
60분
- 2-4. 프로젝트 제출

« 루브릭



아래의 기준을 바탕으로 프로젝트를 평가합니다.

## 평가문항

## 상세기준

1. 적합한 로스와 메트릭을 사용하여 훈련이 이루어졌는가?

데이터셋 구성, 모델 훈련, 결과를 시각화의 한 사이클이 정상적으로 수행되어 테스트 결과를 출력하였다

2. 두 가지 이상의 차이점을 두어 비교가 이루어졌는가?

선택한 모델의 훈련에 필요한 하이퍼 파라미터들의 수치별 성능과 비용의 비교분석을 진행하였다

3. 훈련 결과 및 제품화 가능성에 대한 탐색이 이루어졌는가?

데이터와 모델에 대한 구성 및 훈련 비용과 성능, 모델을 제품화한다면 응용분야와 강점 및 개선사항 대해 정량적, 정성적 분석을 진행하였다

## 프로젝트 제출

프로젝트 제출 방법 안내

- ① 파일 업로드 또는 URL 입력 후 아래의 성취하기 버튼을 클릭하시면 프로젝트를 제출합니다.

파일은 ZIP 파일로 압축해서 첨부해 주신 후 하단의 [성취하기] 버튼을 눌러주세요.

URL의 경우, 올바른 GitHub 또는 Google URL을 입력하신 후 하단의 [성취하기] 버튼을 눌러주세요.

 URL  ZIP 파일ex) <https://github.com/example/url>

&lt; 이전

성취하기 &gt;



아이펠

아이펠 캠퍼스 아이펠 내일배움클래스



## (주)모두의연구소 사업자 정보

대표 김승일 | 개인정보보호책임자 안우진 | 서울특별시 강남구 강남대로 324 역삼디오슈페리움 2층 | cs@aiffel.io |  
 사업자등록번호: 517-88-00184 | 통신판매업신고: 2017-서울강남-04920 | 고객센터: 070 - 7743 - 5882 | 주식회사  
 모두의연구소는 전자상거래 등에서의 소비자보호에 관한 법률에 따른 동신판매업을 영위하고 있습니다.

Copyright ©2024 AIFFEL. All Rights Reserved.