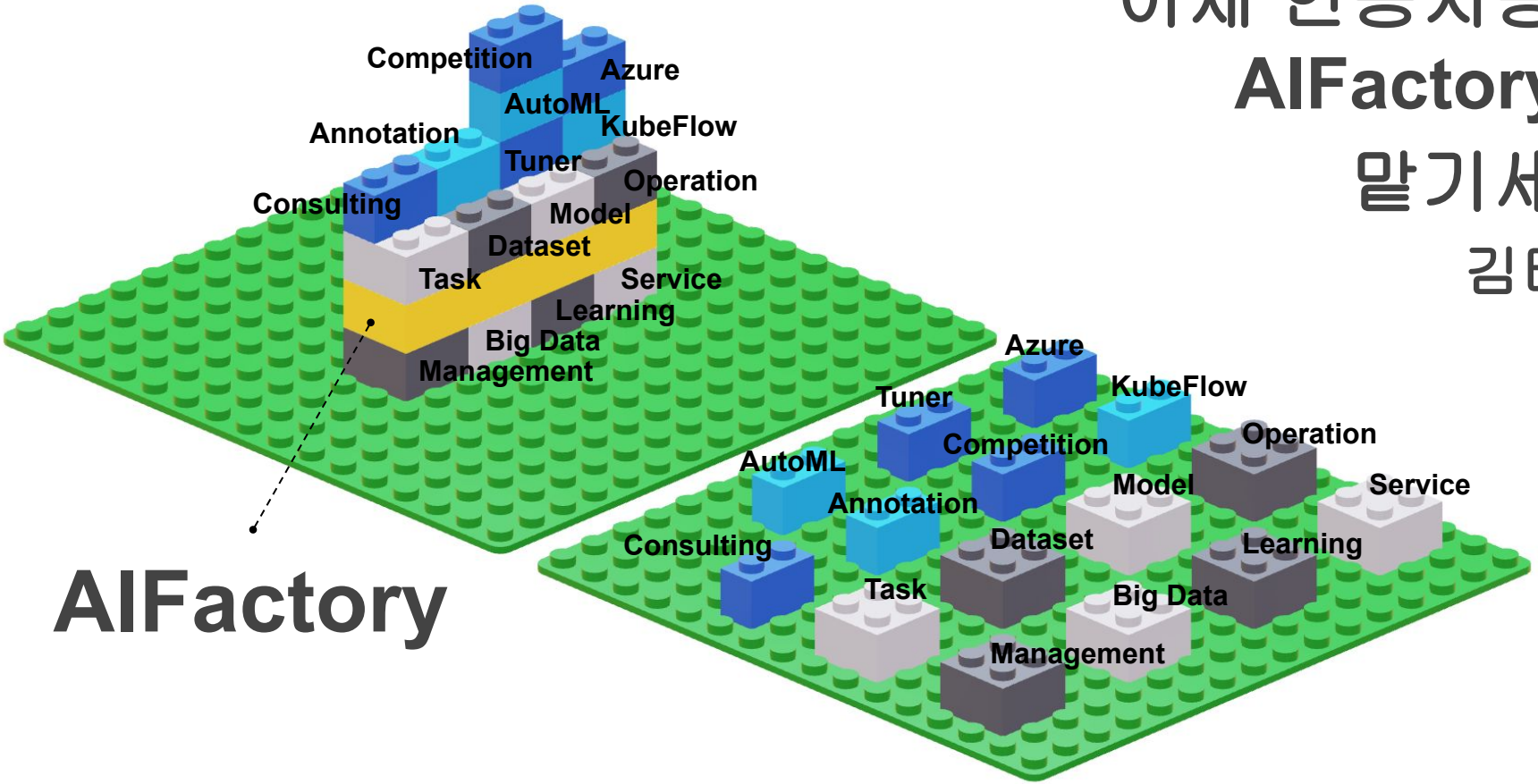
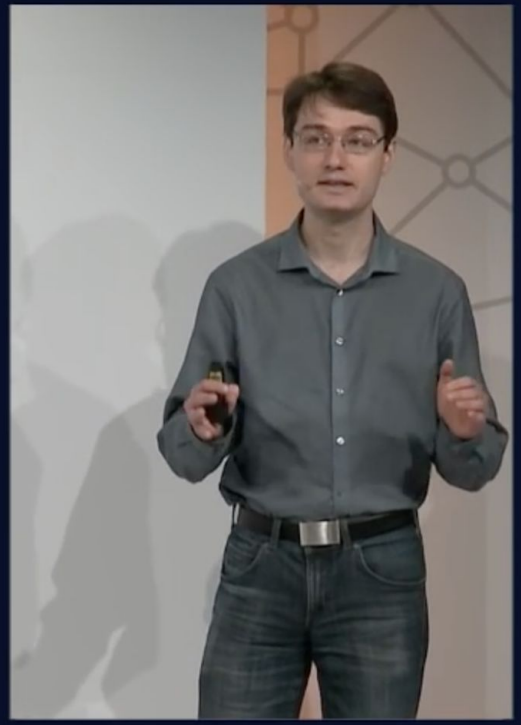
이제 인공지능은 AIFactory에 맡기세요

김태영



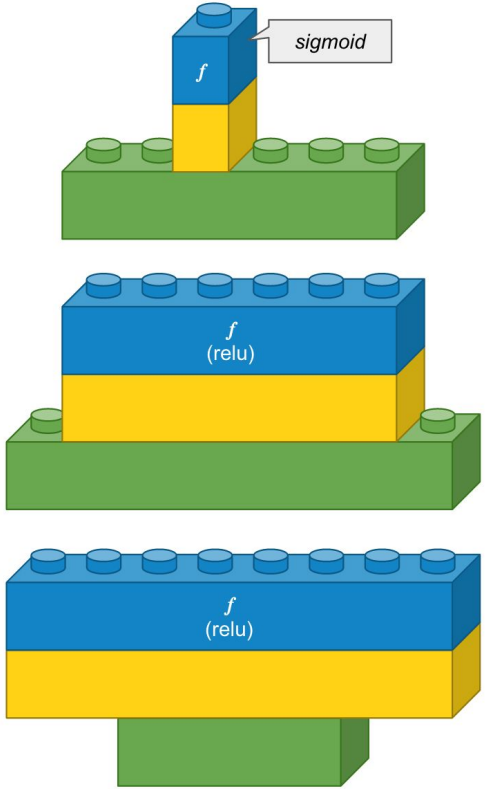


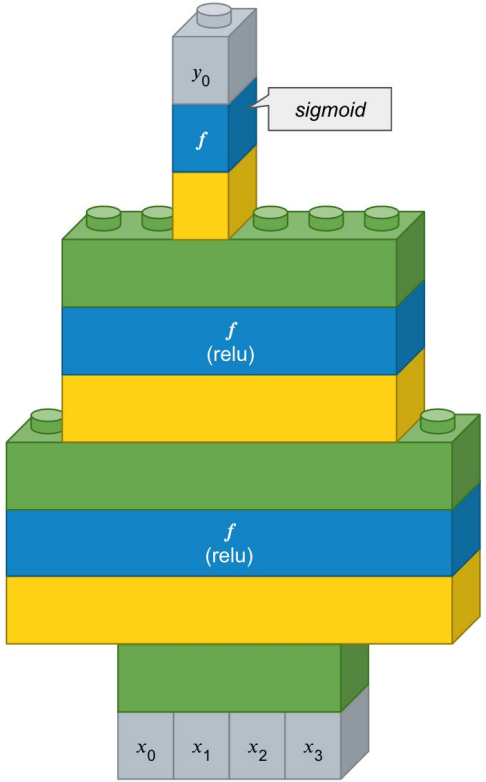
Keras

An API spec for building deep learning

딥러닝을 사용하기 쉽게 만들면 더 많은 분들이 접근할 수 있겠죠. 케라스의 핵심은 모두의 손에 쥐어주는 것입니다.

다층 퍼셉트론 레이어 이야기





DeepBrick for Keras (케라스를 위한 딥브릭)

Sep 10, 2017 • 김태영 (Taeyoung Kim)

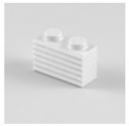
The Keras is a high-level API for deep learning model. The API is very intuitive and similar to building bricks. So, I have started the DeepBrick Project to help you understand Keras's layers and models.

딥러닝과 케라스를 공부하면서 느낀 점은 층을 쌓고 모델을 만들고 하는 과정들이 블록 쌓는 것과 비슷한 느낌을 많이 받았고, 실제로 딥러닝 모델을 설명할 때 블록 그림을 많이 이용하기도 했습니다. 그러다가 (실제 혹은 웹에서) 블록을 쌓으면 딥러닝 모델까지 자동으로 만들 수 있겠다는 생각이 들었습니다. 그래서 딥브릭(DeepBrick)이란 이름으로 프로젝트를 진행해볼까 합니다.

Bricks

There are bricks supported by DeepBrick.

Dataset

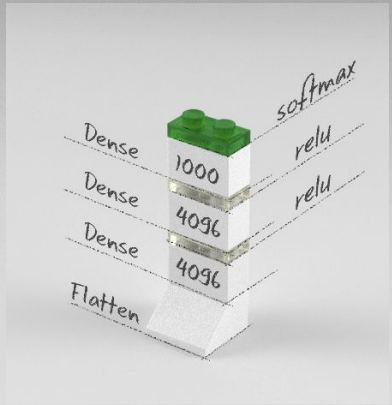
Brick	Name	Description
	Input data, Labels	Input data and labels are encoded as vector. 1차원의 입력 데이터 및 라벨입니다.
	2D Input data	Input data are encoded as 2D vector. 2차원의 입력 데이터입니다. In case of imagery, the dimention consists of sample, width, height and channel. 주로 영상 데이터를 의미하며 샘플수, 너비, 높이, 채널수로 구성됩니다.

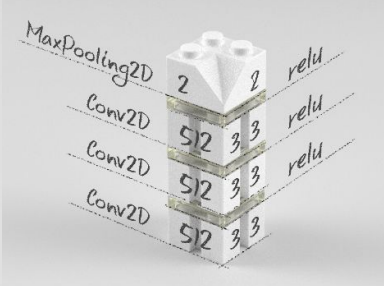
Layers

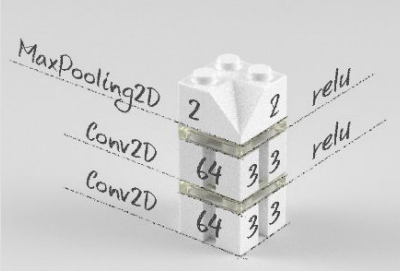
Brick	Name	Description
	Dense	Regular densely-connected neual network layer. 모든 입력 뉴런과 출력 뉴런을 연결하는 전결합층입니다.
	Embedding	Turns positive integer representations of words into a word embedding.
	Dropout	Excludes random input neurons (two-dimensional) at a specified rate during learning to prevent overfitting. 과적합을 방지하기 위해서 학습 시에 지정된 비율만큼 임의의 입력 뉴런(2차원)을 제외시킵니다.

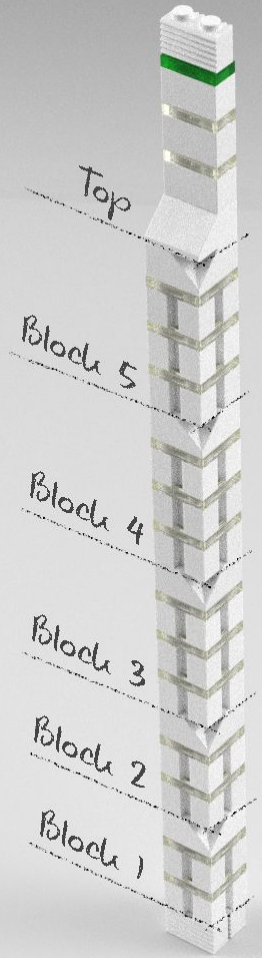
Activation Functions

Brick	Name	Description
	sigmoid	Returns a value between 0 and 1. 활성화 함수로 입력되는 값을 0과 1사이의 값으로 출력시킵니다. This is mainly used for the activation function of the output layer of the binary classification model it can be judged as positive if the output value is above a certain threshold value (for example, 0.5) or negative if it is below. 출력값이 특정 임계값(예를 들어 0.5) 이상이면 양성, 이하이면 음성이라고 판별할 수 있기 때문에 이진분류 모델의 출력층에 주로 사용됩니다.
	softmax	Returns the probability value per class. 활성화 함수로 입력되는 값을 클래스별로 확률 값이 나오도록 출력시킵니다. If all of these probabilities are added, it becomes 1. 이 확률값을 모두 더하면 1이 됩니다. It is used mainly for the activation function of the output layer of a multi-class model, and the class with the highest probability value is the class classified by the model. 다중클래스 모델의 출력층에 주로 사용되며, 확률값이 가장 높은 클래스가 모델이 분류한 클래스입니다.
	tanh	Returns a value between -1 and 1. 활성화 함수로 입력되는 값을 -1과 1사이의 값으로 출력시킵니다. It is used for the activation function of LSTM layer. LSTM의 출력 활성화 함수로 사용됩니다.
	relu	It is mainly used of the activation funition of the hidden layer. 활성화 함수로 주로 은닉층에 사용됩니다.
	relu	It is mainly used of the activation funition of the hidden layer such as Conv2D. 활성화 함수로 주로 Conv2D 은닉층에 사용됩니다.

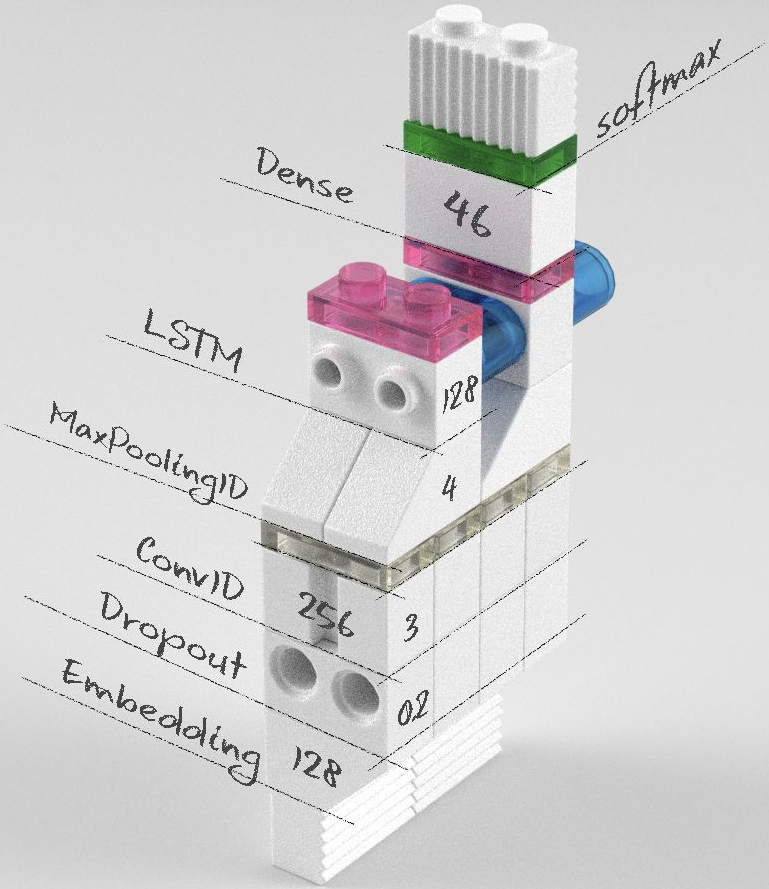


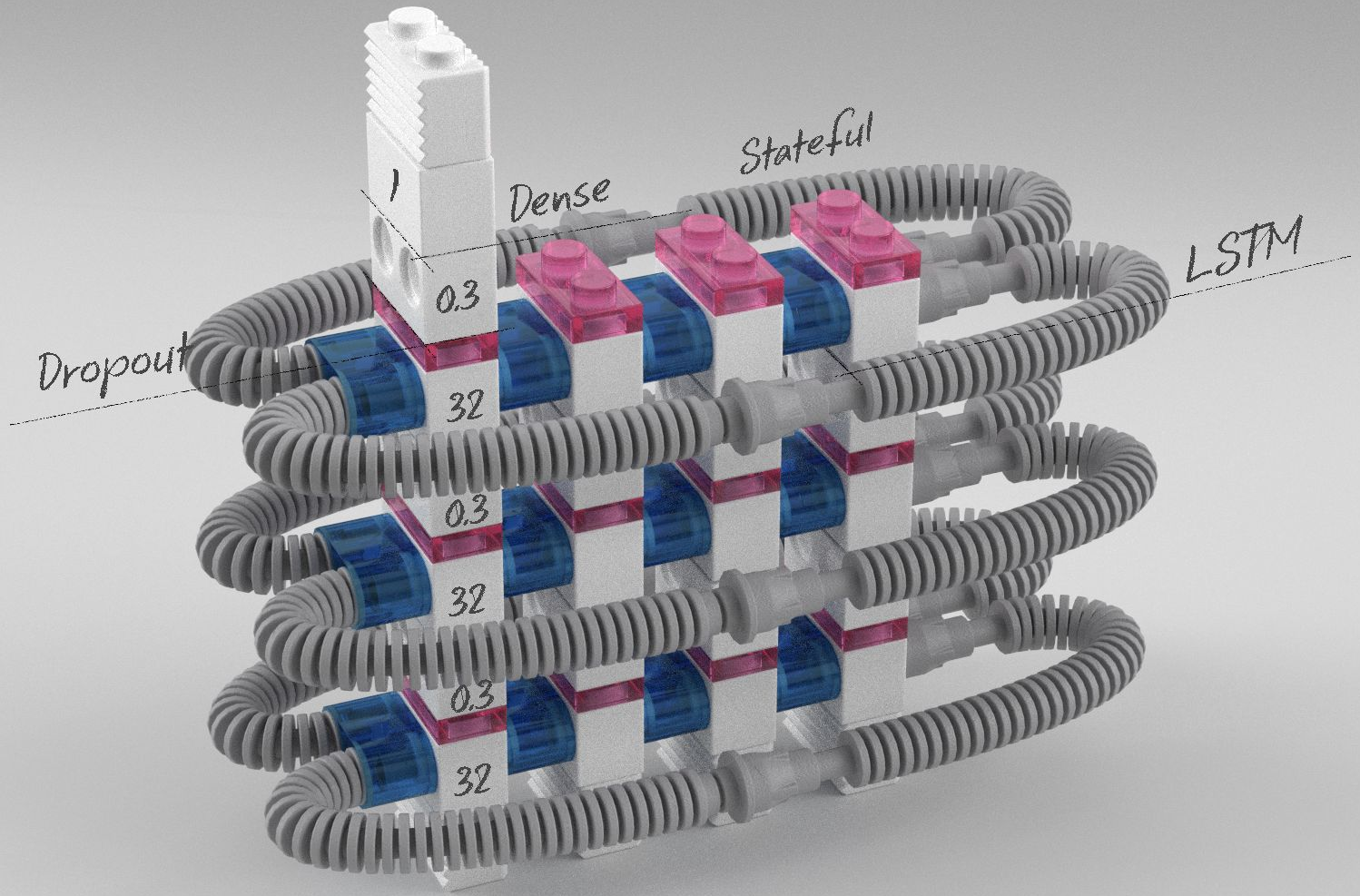


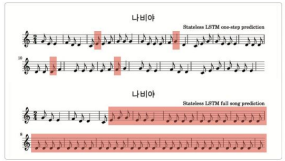





```python
# Block 1
x = Conv2D(64, (3, 3), activation='relu', padding='same',
name='block1_conv1')(img_input)
x = Conv2D(64, (3, 3), activation='relu', padding='same',
name='block1_conv2')(x)
x = MaxPooling2D((2, 2), strides=(2, 2),
name='block1_pool')(x)

# Block 2
x = Conv2D(128, (3, 3), activation='relu', padding='same',
name='block2_conv1')(x)
x = Conv2D(128, (3, 3), activation='relu', padding='same',
name='block2_conv2')(x)
x = MaxPooling2D((2, 2), strides=(2, 2),
name='block2_pool')(x)

# Block 3
x = Conv2D(256, (3, 3), activation='relu', padding='same',
name='block3_conv1')(x)
x = Conv2D(256, (3, 3), activation='relu', padding='same',
name='block3_conv2')(x)
x = Conv2D(256, (3, 3), activation='relu', padding='same',
name='block3_conv3')(x)
x = MaxPooling2D((2, 2), strides=(2, 2),
name='block3_pool')(x)

# Block 4
x = Conv2D(512, (3, 3), activation='relu', padding='same',
name='block4_conv1')(x)
x = Conv2D(512, (3, 3), activation='relu', padding='same',
name='block4_conv2')(x)
x = Conv2D(512, (3, 3), activation='relu', padding='same',
name='block4_conv3')(x)
x = MaxPooling2D((2, 2), strides=(2, 2),
name='block4_pool')(x)

# Block 5
x = Conv2D(512, (3, 3), activation='relu', padding='same',
name='block5_conv1')(x)
x = Conv2D(512, (3, 3), activation='relu', padding='same',
name='block5_conv2')(x)
x = Conv2D(512, (3, 3), activation='relu', padding='same',
name='block5_conv3')(x)
x = MaxPooling2D((2, 2), strides=(2, 2),
name='block5_pool')(x)

if include_top:
    # Classification block
    x = Flatten(name='flatten')(x)
    x = Dense(4096, activation='relu', name='fc1')(x)
    x = Dense(4096, activation='relu', name='fc2')(x)
    x = Dense(classes, activation='softmax',
name='predictions')(x)
else:
    if pooling == 'avg':
        x = GlobalAveragePooling2D()(x)
    elif pooling == 'max':
        x = GlobalMaxPooling2D()(x)
```


softmax
Dense
46
LSTM
128
MaxPooling1D
4
Conv1D
256
3
Dropout
02
Embedding
128


1
Dense
Stateful
LSTM
Dropout
0.3
32
0.3
32
0.3
32



위 악보로 연주한 곡은 아래 링크에서 다운로드할 수 있습니다.
http://tykimos.github.com/Keras/warehouse/2017-4-9-Stateless_LSTM_one_step_prediction_mp3
http://tykimos.github.com/Keras/warehouse/2017-4-9-Stateless_LSTM_full_song_prediction_mp3

7. 상태유지 LSTM 모델

이번에는 상태유지(Stateful) LSTM 모델에 대하여 알아보겠습니다. 여기서 상태유지라는 것은 현재 학습된 상태가 다음 학습 시 초기 상태로 전달된다는 것을 의미합니다.

```
상태유지 모드에서는 현재 샘플의 학습 상태가 다음 샘플의 초기 상태로 전달됩니다.
```

긴 시퀀스 데이터를 처리할 때, LSTM 모델은 상태유지 모드에서 그 전가를 발휘합니다. 긴 시퀀스 데이터를 샘플 단위로 잘라서 학습하더라도 LSTM 네부적으로 기억할 것은 기억하고, 버릴 것은 버려서 기억해야 할 중요한 정보만 이어갈 수 있도록 상태가 유지되기 때문입니다. 상태유지 LSTM 모델을 생성하기 위해서는 LSTM 레이어 생성 시, stateful=True로 설정합니다. 또한 상태유지 모드에서는 입력형태를 batch_input_shape = (배치사이즈, 타임스텝, 속성)으로 설정해야 합니다. 상태유지 모드에서 배치사이즈 개념은 조금 이따주므로 다음 장에서 다루기로 하겠습니다.

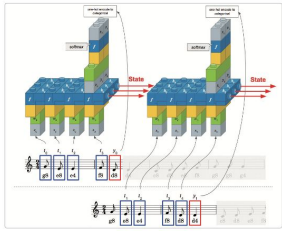
```
model = Sequential()
model.add(LSTM(128, batch_input_shape = (1, 4, 1), stateful=True))
model.add(Dense(one_hot_vec_size, activation='softmax'))
```


상태유지 모드에서는 모델 학습 시, 상태 초기화에 대한 고민이 필요합니다. 현재 샘플 학습 상태가 다음 샘플 학습의 초기상태로 상태를 유지시키지 않고 초기화해야 합니다. 전달되는 시인데, 현재 샘플과 다음 샘플 간의 순차적인 관계가 없을 경우에는 상태가 유지되지 않고 초기화가 되어야 합니다. 예를 들면,

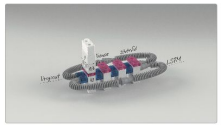
- 마지막 샘플 학습을 마치고, 새로운 에포크 수행 시 새로운 샘플을 학습을 해야하므로 상태 초기화 필요
- 한 에포크 안에 여러 시퀀스 데이터 세트가 있을 경우, 새로운 시퀀스 데이터 세트를 학습 전에 상태 초기화 필요

현재 코드에서는 한 곡을 가지고 재속 학습을 시키고 있으므로 새로운 에포크 시작 시에만 상태 초기화를 수행하면 됩니다.

```
run_epochs = 2000

for epoch_idx in range(run_epochs):
    print ('epochs :' + str(epoch_idx) )
    model.fit(x_train, y_train, epochs=1, batch_size=1, verbose=2, shuffle=False) # 50 is
X_shape[0]
    model.reset_states()
```



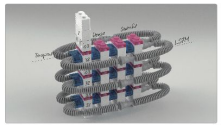

```
model.add(Dropout(0.3))
model.add(Dense(1))
```



- **상태유지 스택 순환신경망 모델**

상태유지 순환신경망을 여러 층 쌓아올린 모델입니다. 층이 하나인 순환신경망에 비해 더 깊은 추론이 가능한 모델입니다.

```
model = Sequential()
for i in range(2):
    model.add(LSTM(32, batch_input_shape=(1, look_back, 1), stateful=True, return_
sequences=True))
    model.add(Dropout(0.3))
model.add(LSTM(32, batch_input_shape=(1, look_back, 1), stateful=True))
model.add(Dropout(0.3))
model.add(Dense(1))
```



이 표현은 다음을 의미합니다.

- **입력 이미지 사이즈가 3x3이고 채널이 3개입니다.**
- **2x2 커널을 가진 필터가 2개입니다. 채널마다 커널이 합업되어 총 기중치는 3x2x2x2로 24개 입니다.**
- **출력 이미지는 사이즈가 3x3이고 채널이 2개입니다.**

2. 사소한 변화를 무시해주는 맥스풀링(Max Pooling) 레이어

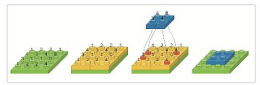
컨볼루션 레이어의 출력 이미지에서 주요값만 뽑아 크기가 작은 출력 영상을 만듭니다. 이것은 지역적인 사소한 변화가 영향을 미치지 않도록 합니다.

```
MaxPooling2D(pool_size=(2, 2))
```

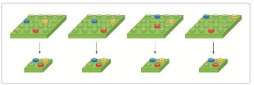
주요 인자는 다음과 같습니다.

- **pool_size :** 수직, 수평 축소 비율을 지정합니다. (2, 2)이면 출력 영상 크기는 입력 영상 크기의 반으로 줄어듭니다.

예를 들어, 입력 영상 크기가 4x4이고, pool_size를 (2, 2)로 했을 때를 도식화하면 다음과 같습니다. 녹색 블록은 입력 영상을 나타내고, 노란색 블록은 pool_size에 따라 나눈 경계를 표시합니다. 해당 경계에서 가장 큰 값을 선택하여 작은 블록으로 만들면, 그것이 출력 영상이 됩니다. 가장 오른쪽은 맥스풀링 레이어를 약식으로 표시한 것입니다.



이 레이어는 영상의 작은 변화라든지 시소한 움직임이 특징을 추출할 때 크게 영향을 미치지 않도록 합니다. 영상 내에 특징이 세 개가 있다고 가정했을 때, 아래 그림에서 첫 번째 영상을 기준으로 두 번째 영상은 오른쪽으로 이동하였고, 세 번째 영상은 약간 비틀어 졌고, 네 번째 영상은 조금 확대되었습니다. 맥스풀링한 결과는 모두 동일합니다. 일곱 인식 문제를 예를 들면, 맥스풀링의 역할은 사람이다 는, 코, 입 위치가 조금씩 다른데 이러한 차이가 사람이라고 인식하는데 있어서는 큰 영향을 미치지 않게 합니다.



3. 영상을 일차원으로 바꿔주는 플래튼(Flatten) 레이어

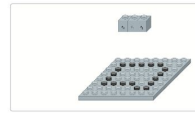
컨볼루션 신경망 모델에서 컨볼루션 레이어나 맥스풀링 레이어를 반복적으로 거치면 주요 특징만 추출되고 추출된 주요 특징은 전결합층에 전달되어 학습합니다. 컨볼루션 레이어나 맥스풀링 레이어는 주요 2차원 자료를 다루지만 전결합층에 전달하기 위해선 1차원 자료로 바꿔줘야 합니다. 이 때 사용되는 것이 플래튼 레이어입니다. 사용 예시는 다음과 같습니다.

```
Flatten()
```

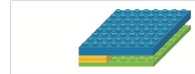
이전 레이어의 출력 정보를 이용하여 입력 정보는 자동으로 설정되며, 출력 형태는 입력 형태에 따라 자동으로 계산되기 때문에 별도로 사용자가 파라미터를 지정하주지 않아도 됩니다. 크기가 3x3인 영상을 1차원으로 변경할을 경우를 도식화하면 다음과 같습니다.

4. 한 번 쌓아보기

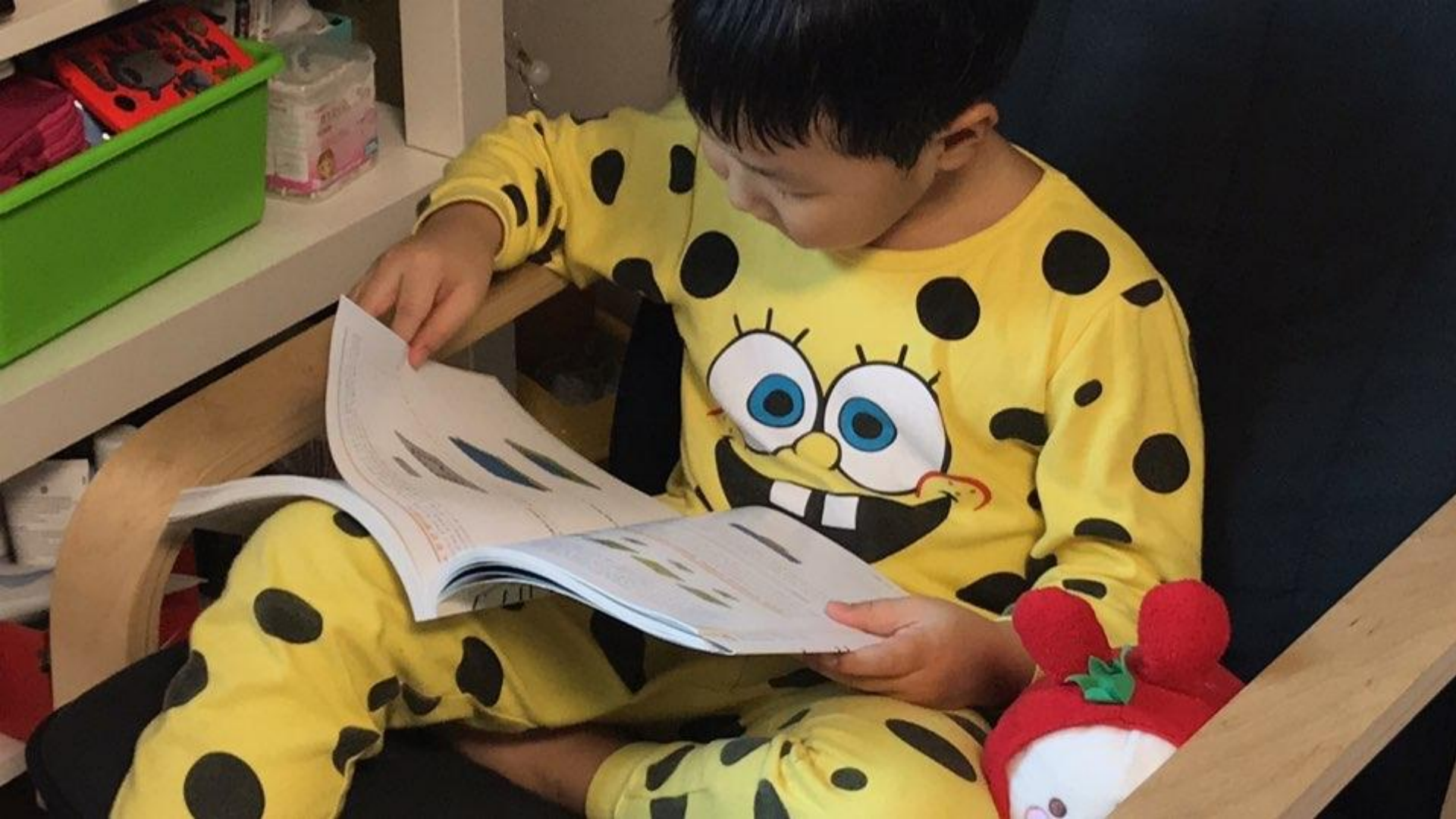
지금까지 알아본 레이어를 이용해서 간단한 컨볼루션 신경망 모델을 만들어보겠습니다. 먼저 간단한 문제를 정해봅시다. 삼각형, 사각형, 원을 손으로 그린 이미지가 있고 이미지 크기가 8x8이며 그 가중치들니다. 삼각형, 사각형, 원을 구분하는 3개의 클래스를 분류하는 문제이며 제문에 출력 뉴런는 3개여야 합니다. 필요하다고 생각하는 레이어를 구성해보았습니다.

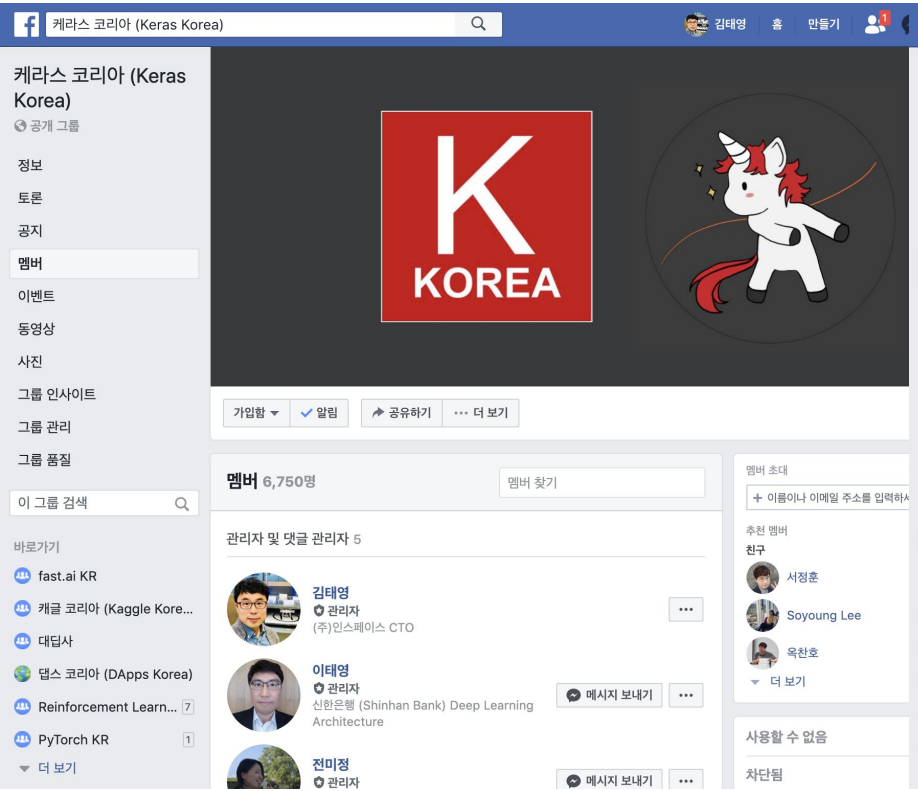


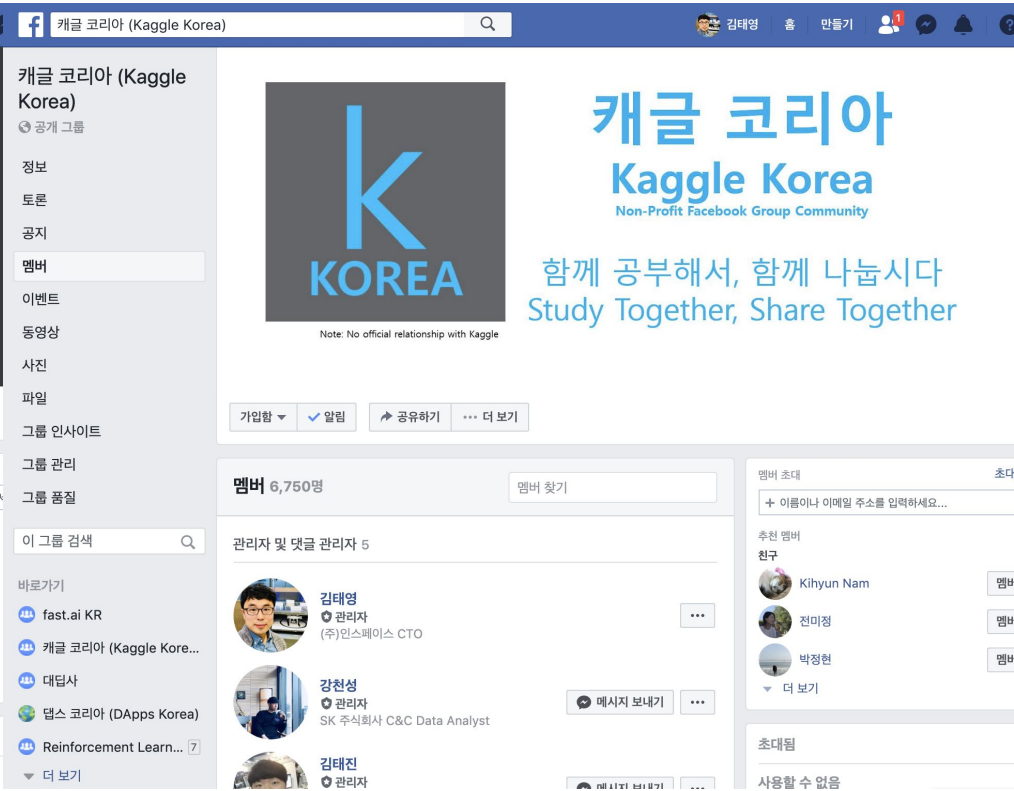
- **컨볼루션 레이어 :** 입력 이미지 크기 8x8, 입력 이미지 채널 1개, 필터 크기 3x3, 필터 수 2개, 경계 타입 'same', 활성화 함수 'relu'



- **맥스풀링 레이어 :** 풀 크기 2x2




케라스 코리아 (Keras Korea)
김태영
홈
만들기
케라스 코리아 (Keras Korea)
공개 그룹
정보
토론
공지
멤버
이벤트
동영상
사진
그룹 인사이트
그룹 관리
그룹 품질
이 그룹 검색
바로가기
fast.ai KR
캐글 코리아 (Kaggle Kore...
대딥사
댑스 코리아 (DApps Korea)
Reinforcement Learn... 7
PyTorch KR 1
더 보기
K
KOREA
가입함
알림
공유하기
더 보기
멤버 6,750명
멤버 찾기
관리자 및 댓글 관리자 5
김태영
관리자
(주)인스페이스 CTO
이태영
관리자
신한은행 (Shinhan Bank) Deep Learning Architecture
메시지 보내기
전미정
관리자
메시지 보내기
멤버 초대
이름이나 이메일 주소를 입력하세...
추천 멤버
친구
서정훈
Soyoung Lee
옥찬호
더 보기
사용할 수 없음
차단됨


캐글 코리아 (Kaggle Korea)
김태영
홈
만들기
캐글 코리아 (Kaggle Korea)
공개 그룹
정보
토론
공지
멤버
이벤트
동영상
사진
파일
그룹 인사이트
그룹 관리
그룹 품질
이 그룹 검색
바로가기
fast.ai KR
캐글 코리아 (Kaggle Kore...
대딥사
댑스 코리아 (DApps Korea)
Reinforcement Learn... 7
더 보기
k
KOREA
Note: No official relationship with Kaggle
캐글 코리아
Kaggle Korea
Non-Profit Facebook Group Community
함께 공부해서, 함께 나눕시다
Study Together, Share Together
가입함
알림
공유하기
더 보기
멤버 6,750명
멤버 찾기
관리자 및 댓글 관리자 5
김태영
관리자
(주)인스페이스 CTO
강천성
관리자
SK 주식회사 C&C Data Analyst
메시지 보내기
김태진
관리자
멤버 초대
이름이나 이메일 주소를 입력하세요...
추천 멤버
친구
Kihyun Nam
전미정
박정현
더 보기
초대됨
사용할 수 없음

nature astronomy

LETTERS

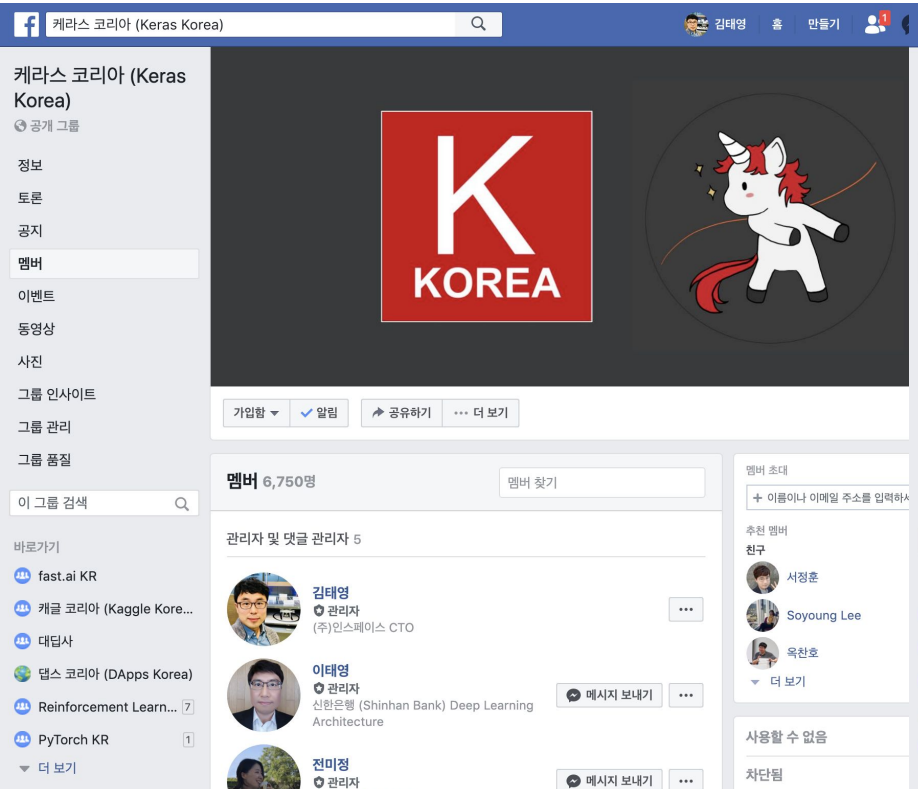
https://doi.org/10.1038/s41550-019-0711-5

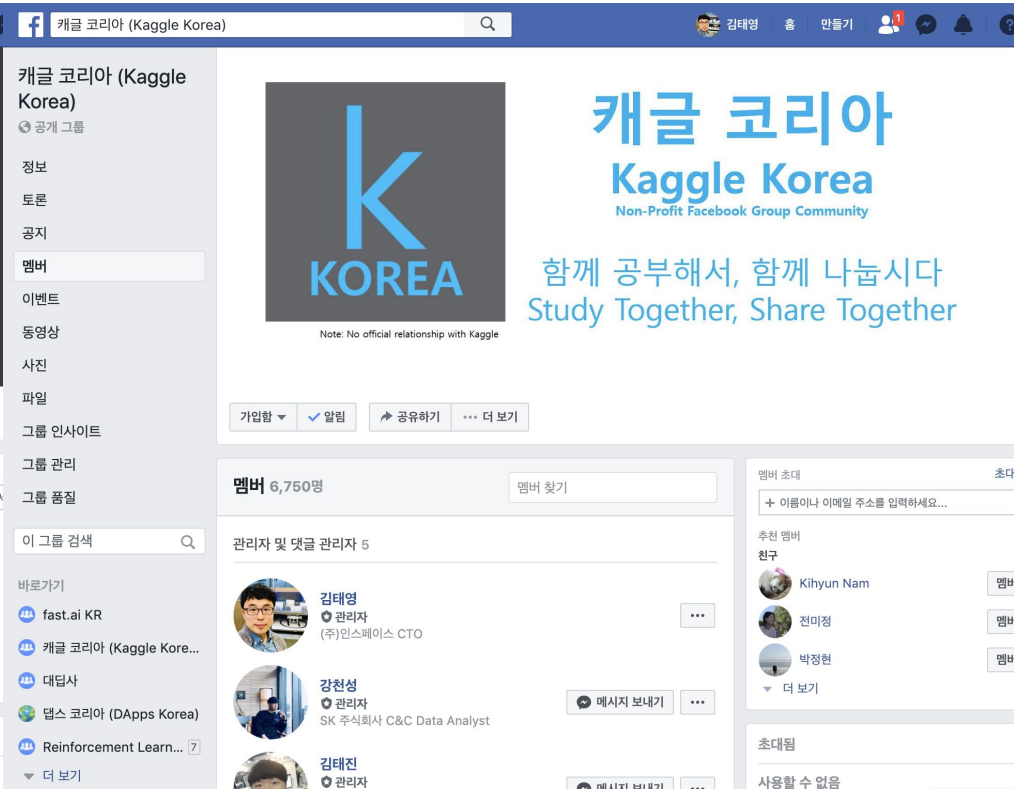
Solar farside magnetograms from deep learning analysis of STEREO/EUVI data

Taeyoung Kim[1,7], Eunsu Park[1,7], Harim Lee[1,2,7], Yong-Jae Moon[1,2*], Sung-Ho Bae[3], Daye Lim[1], Soojeong Jang[4], Lokwon Kim[3], Il-Hyun Cho[2], Myungjin Choi[5] and Kyung-Suk Cho[4,6]

Solar magnetograms are important for studying solar activity and predicting space weather disturbances[1]. Farside magnetograms can be constructed from local helioseismology without any farside data[2-4], but their quality is lower than that of typical frontside magnetograms. Here we generate farside solar magnetograms from STEREO/Extreme UltraViolet Imager (EUVI) 304-Å images using a deep learning model based on conditional generative adversarial networks (cGANs). We train the model using pairs of Solar Dynamics Observatory (SDO)/Atmospheric Imaging Assembly (AIA) 304-Å images

training step, the generator is trained to learn the polarity patterns of active regions. In the evaluation and generation step, the generator reproduces the pattern. Since all data are from the 24th solar cycle, there is no difficulty in producing the Hale's law pattern in this cycle. We note that the polarity of the solar magnetic field is reversed cycle by cycle. Hence, since our model has been trained on the 24th solar cycle, it would be effective for even solar cycles, but should be tested for odd cycles. A careful comparison between two magnetograms shows that the tilt angle between a preceding sunspot and the one that follows it is not always properly generated,


케라스 코리아 (Keras Korea)
김태영
홈
만들기
케라스 코리아 (Keras Korea)
공개 그룹
정보
토론
공지
멤버
이벤트
동영상
사진
그룹 인사이트
그룹 관리
그룹 품질
이 그룹 검색
바로가기
fast.ai KR
캐글 코리아 (Kaggle Kore...
대딥사
댑스 코리아 (DApps Korea)
Reinforcement Learn... 7
PyTorch KR 1
더 보기
K
KOREA
가입함
알림
공유하기
더 보기
멤버 6,750명
멤버 찾기
관리자 및 댓글 관리자 5
김태영
관리자
(주)인스페이스 CTO
이태영
관리자
신한은행 (Shinhan Bank) Deep Learning Architecture
메시지 보내기
전미정
관리자
메시지 보내기
멤버 초대
이름이나 이메일 주소를 입력하세...
추천 멤버
친구
서정훈
Soyoung Lee
옥찬호
더 보기
사용할 수 없음
차단됨

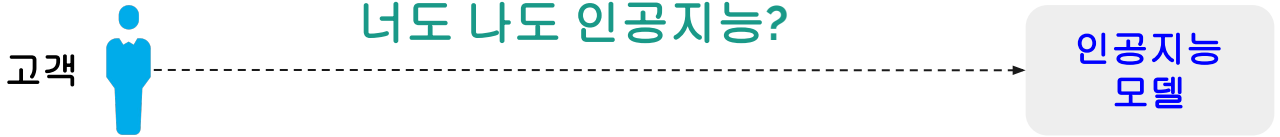

캐글 코리아 (Kaggle Korea)
김태영
홈
만들기
캐글 코리아 (Kaggle Korea)
공개 그룹
정보
토론
공지
멤버
이벤트
동영상
사진
파일
그룹 인사이트
그룹 관리
그룹 품질
이 그룹 검색
바로가기
fast.ai KR
캐글 코리아 (Kaggle Kore...
대딥사
댑스 코리아 (DApps Korea)
Reinforcement Learn... 7
더 보기
k
KOREA
Note: No official relationship with Kaggle
캐글 코리아
Kaggle Korea
Non-Profit Facebook Group Community
함께 공부해서, 함께 나눕시다
Study Together, Share Together
가입함
알림
공유하기
더 보기
멤버 6,750명
멤버 찾기
관리자 및 댓글 관리자 5
김태영
관리자
(주)인스페이스 CTO
강천성
관리자
SK 주식회사 C&C Data Analyst
메시지 보내기
김태진
관리자
멤버 초대
이름이나 이메일 주소를 입력하세요...
추천 멤버
친구
Kihyun Nam
전미정
박정현
더 보기
초대됨
사용할 수 없음

비지니스 모델

제조 혁신 >> Smart Factory

AI 혁신 >> AI Factory

배경



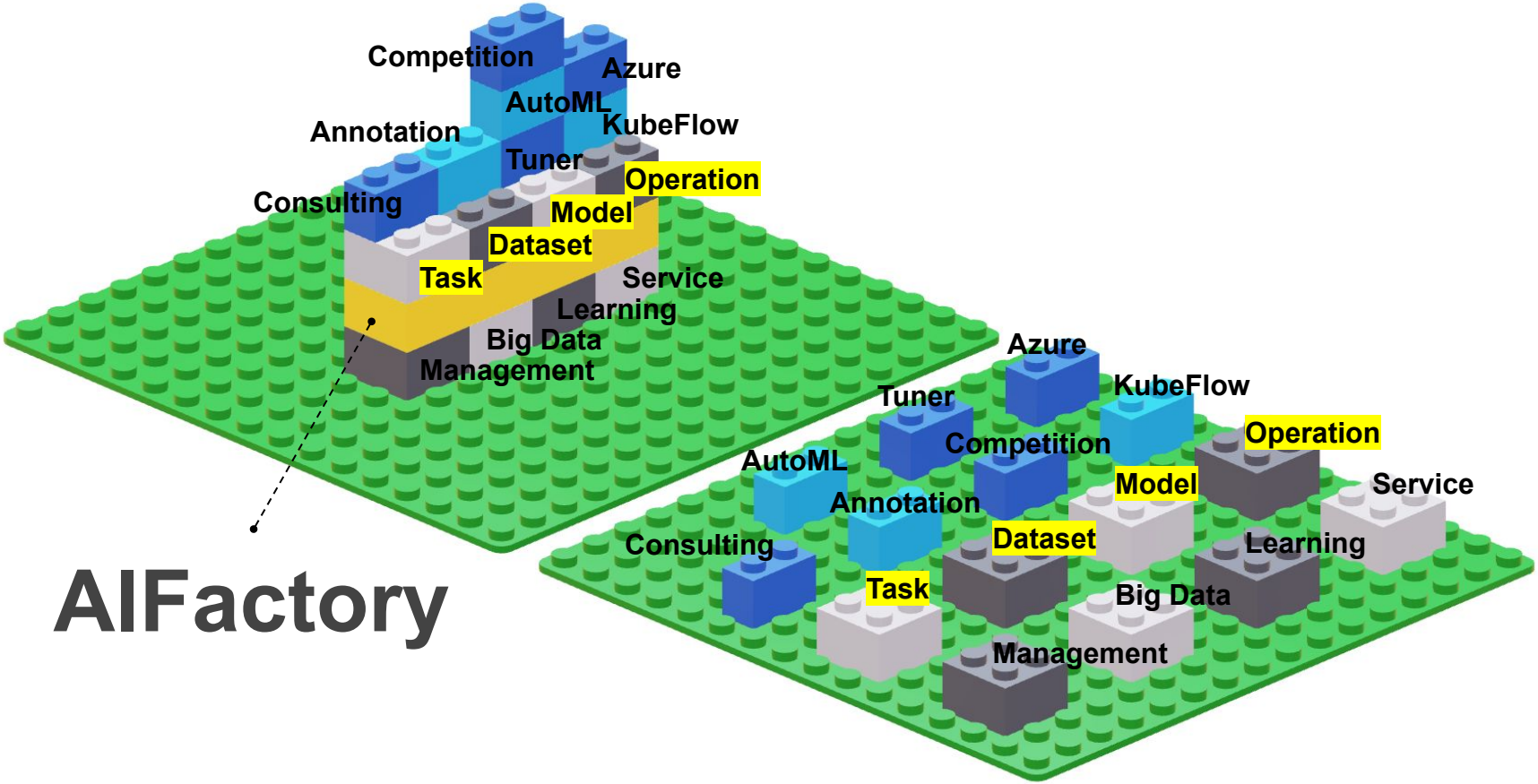
인공지능 기술
수요 급증

- 인공지능의 눈부신 발전속도 >> 더불어 오픈되는 소스코드
- 많은 튜토리얼과 다양한 적용사례 정보 >> 기술접근 용이
- 각종 정부지원 사업 및 범용 인공지능 모델 서비스 >> 기술 보편화

하지만, 세가지 요소를 모두 갖춰야만 인공지능 기술 도입이 가능

애로사항

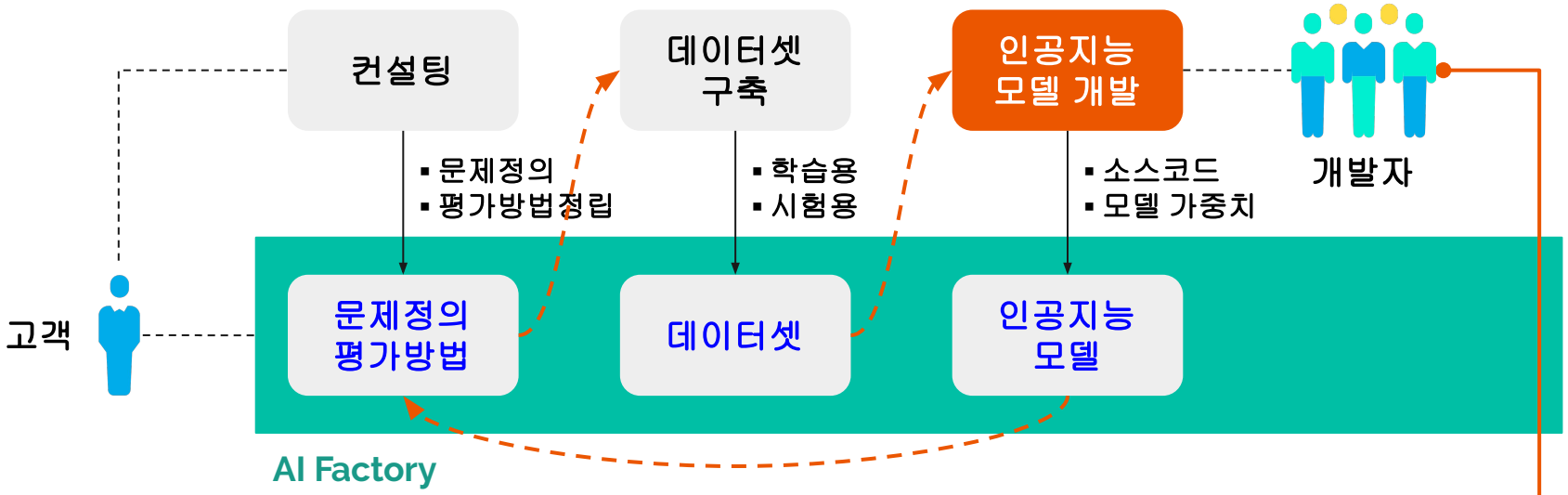
- 인공지능 기술의 높은 진입장벽 >> 많은 선행 지식 필요
- 데이터 중심적 모델링 방법 >> 나만의 모델이 필요
- 인력, 컴퓨팅 자원 제약 >> 단편적인 시도에 만족해야 함
- 인공지능 전문가 품귀현상 >> 높은 비용
- 성능 예측의 어려움 >> 높은 리스크
- 소요시간 산정의 어려움 >> 시장 선점 실패


Competition
Azure
AutoML
KubeFlow
Annotation
Tuner
Operation
Consulting
Model
Dataset
Task
Service
Learning
Big Data
Management
Azure
KubeFlow
Tuner
Operation
Competition
AutoML
Model
Service
Annotation
Dataset
Learning
Consulting
Task
Big Data
Management

AIFactory

비지니스 모델 : 인공지능 모델 개발

크라우드 소싱



크라우드 소싱 기반 인공지능 문제해결 협업 플랫폼

- 세 가지 요소를 플랫폼에서 묶어줌 >> 고객은 컨설팅만 수행
- 목표 성능 선착순 달성 보상 지급 방식
  >> 기간과 비용을 단축시키면서 고성능의 모델 획득

비지니스 모델 : 인공지능 모델 개발

넷플릭스 추천시스템
머신러닝 경연대회

상금
$1,000,000
(약 11.7억원)

빅데이터 분석 분야에
크라우드 소싱 가능성
입증

넷플릭스 핵심인
추천시스템
알고리즘의
발전 계기가 됨

4회	2달	6시간	1억	92%	-
랩	기간	체크포인트	상금	목표	달성
1회	2주	-	3천	-	-
2회	-	6시간	-	-	-
3회	-	6시간	-	-	-
4회	-	6시간	-	92%	-

	랩	기간	체크포인트	상금	목표	달성
	4회	2달	6시간	1억	92%	-
시작	1회	2주	-	3천	-	-
	2회	-	6시간	-	-	-
	3회	-	6시간	-	-	-
	4회	-	6시간	-	92%	-

	랩	기간	체크포인트	상금	목표	달성
	4회	2달	6시간	1억	92%	-
2주 후	1회	2주	-	3천	-	70%
	2회	-	6시간	-	-	-
	3회	-	6시간	-	-	-
	4회	-	6시간	-	92%	-

	랩	기간	체크포인트	상금	목표	달성
	4회	2달	6시간	1억	92%	-
	1회	2주	-	3천	-	70%
랩설정	2회	-	6시간	2천	80%	-
	3회	-	6시간	2천	88%	-
	4회	-	6시간	3천	92%	-

	랩	기간	체크포인트	상금	목표	달성
	4회	2달	6시간	1억	92%	-
	1회	2주	-	3천	-	70%
시작	2회	-	6시간	2천	80%	-
	3회	-	6시간	2천	88%	-
	4회	-	6시간	3천	92%	-

	랩	기간	체크포인트	상금	목표	달성
	4회	2달	6시간	1억	92%	-
	1회	2주	-	3천	-	70%
3일 후	2회	3일	6시간	2천	80%	89%
	3회	-	6시간	2천	88%	-
	4회	-	6시간	3천	92%	-

	랩	기간	체크포인트	상금	목표	달성
	4회	2달	6시간	1억	92%	-
	1회	2주	-	3천	-	70%
	2회	3일	6시간	2천	80%	89%
재설정	3회	-	6시간	2천	92%	-
	4회	-	6시간	3천	95%	-

	랩	기간	체크포인트	상금	목표	달성
	4회	2달	6시간	1억	92%	-
	1회	2주	-	3천	-	70%
	2회	3일	6시간	2천	80%	89%
시작	3회	-	6시간	2천	92%	-
	4회	-	6시간	3천	95%	-

	랩	기간	체크포인트	상금	목표	달성
	4회	2달	6시간	1억	92%	-
	1회	2주	-	3천	-	70%
	2회	3일	6시간	2천	80%	89%
1주 후	3회	1주	6시간	2천	92%	94%
	4회	-	6시간	3천	95%	-

	랩	기간	체크포인트	상금	목표	달성
	4회	2달	6시간	1억	92%	-
	1회	2주	-	3천	-	70%
	2회	3일	6시간	2천	80%	89%
	3회	1주	6시간	2천	92%	94%
재설정	4회	-	6시간	3천	97%	-

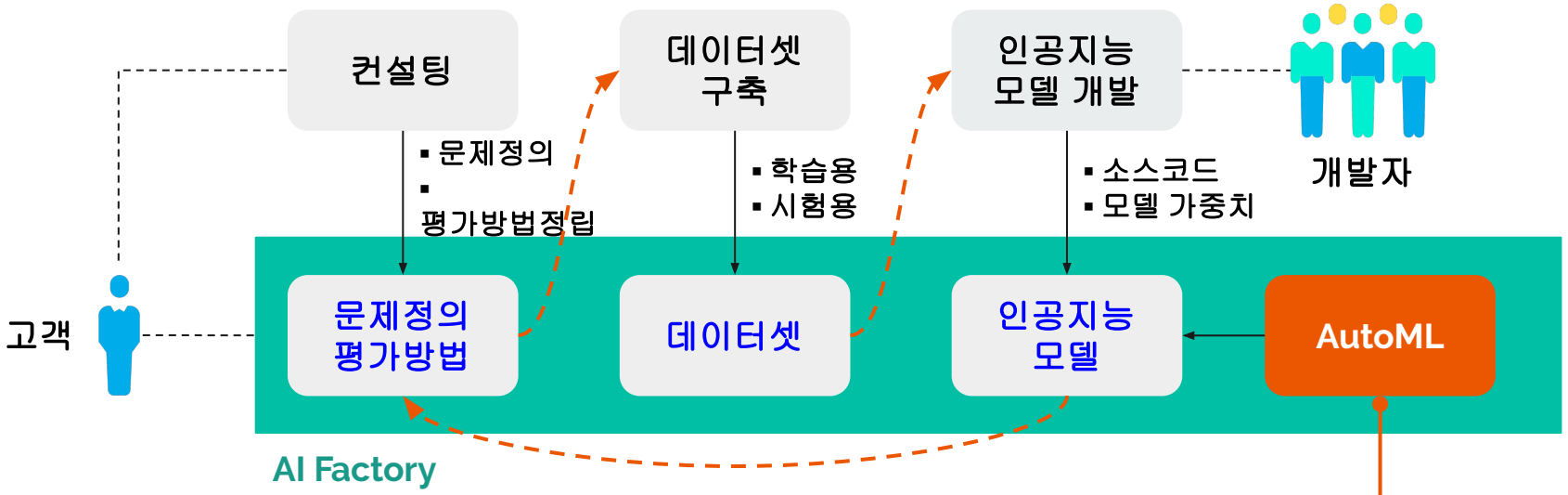
	랩	기간	체크포인트	상금	목표	달성
	4회	2달	6시간	1억	92%	-
	1회	2주	-	3천	-	70%
	2회	3일	6시간	2천	80%	89%
	3회	1주	6시간	2천	92%	94%
시작	4회	-	6시간	3천	97%	-

	랩	기간	체크포인트	상금	목표	달성
	4회	2달	6시간	1억	92%	-
	1회	2주	-	3천	-	70%
	2회	3일	6시간	2천	80%	89%
	3회	1주	6시간	2천	92%	94%
4일 후	4회	4일	6시간	3천	97%	98%

4회	2달	6시간	1억	92%	98%
랩	기간	체크포인트	상금	목표	달성
1회	2주	-	3천	-	70%
2회	3일	6시간	2천	80%	89%
3회	1주	6시간	2천	92%	94%
4회	4일	6시간	3천	97%	98%
총 기간	3주				

4회	2달	6시간	1억	92%	98%
랩	기간	체크포인트	상금	목표	달성
1회	2주	-	3천	-	70%
2회	3일	6시간	2천	80%	89%
3회	1주	6시간	2천	92%	94%
4회	4일	6시간	3천	97%	98%
총 기간	3주				

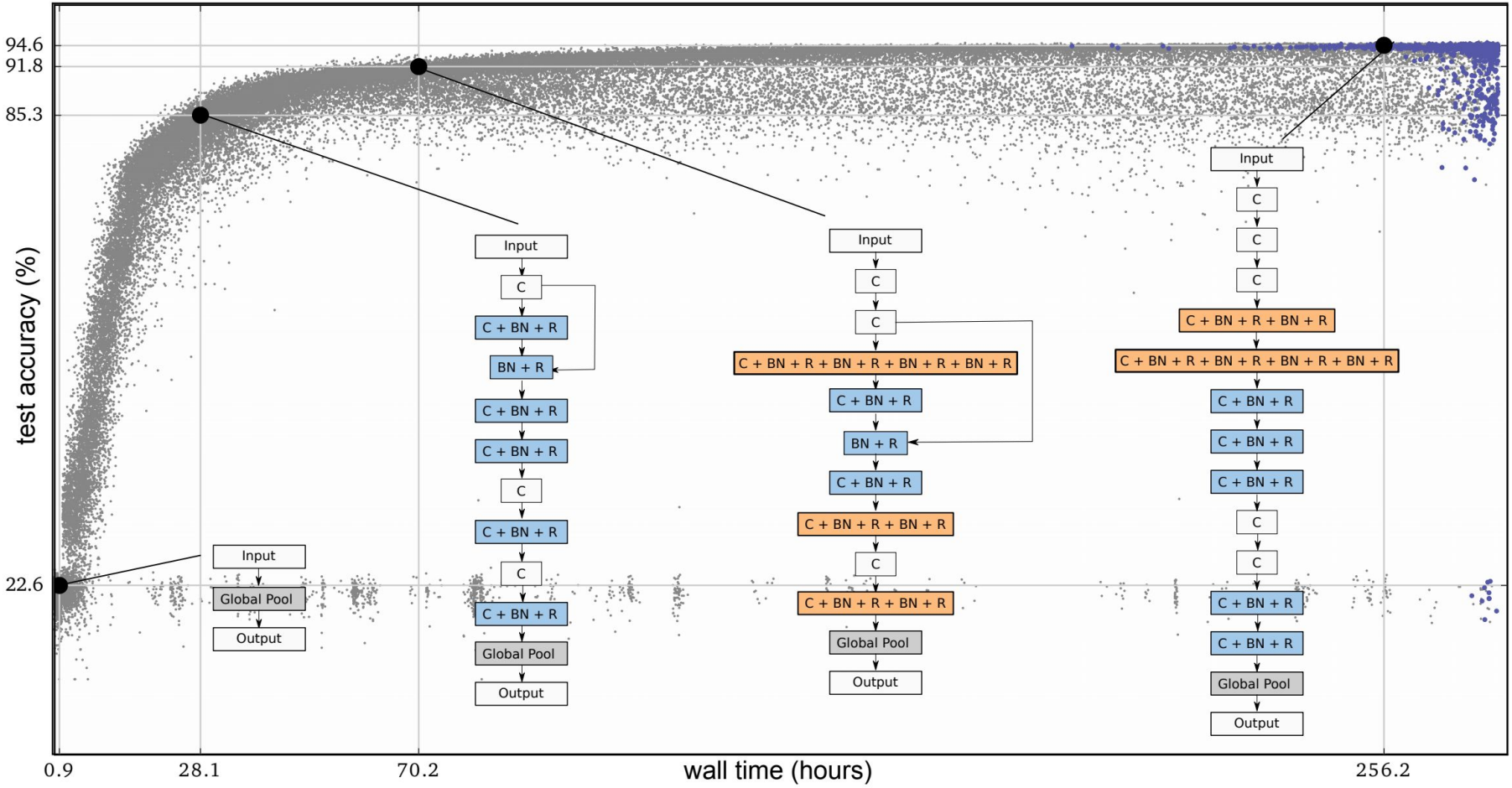
비지니스 모델 : 인공지능 모델 개발



데이터셋에서 머신러닝 모델을 자동으로 생성

- AutoML이 개발자와 함께 경연대회 가능
- AutoML만으로 수행할 경우 데이터 공개 불필요

비지니스 모델 : AutoML



AutoML

AutoKeras

Example

Here is a short example of using the package.

```
import autokeras as ak

clf = ak.ImageClassifier()
clf.fit(x_train, y_train)
results = clf.predict(x_test)
```

For detailed tutorial, please check here.

```
model = Sequential()
model.add(Conv2D(32, kernel_size=(3, 3),
                 activation='relu',
                 input_shape=input_shape))
model.add(Conv2D(64, (3, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))
model.add(Flatten())
model.add(Dense(128, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(num_classes, activation='softmax'))

model.compile(loss=keras.losses.categorical_crossentropy,
              optimizer=keras.optimizers.Adadelta(),
              metrics=['accuracy'])

model.fit(x_train, y_train,
          batch_size=batch_size,
          epochs=epochs,
          verbose=1,
          validation_data=(x_test, y_test))
score = model.evaluate(x_test, y_test, verbose=0)
```

```
clf = ImageClassifier(verbose=True, augment=False)

clf.fit(x_train, y_train, time_limit=12 * 60 * 60)
clf.final_fit(x_train, y_train, x_test, y_test, retrain=True)
y = clf.evaluate(x_test, y_test)

print(y * 100)
print("end")
```

MNIST Test Accuracy

Keras : 0.9912
AutoKeras : 0.994(0.998)

Task API

AutoKeras support the following task APIs.

ImageClassifier class

[source]

```
autokeras.task.ImageClassifier(
    num_classes=None,
    multi_label=False,
    loss=None,
    metrics=None,
    name="image_classifier",
    max_trials=100,
    directory=None,
    objective="val_loss",
    seed=None,
)
```

AutoKeras image classification class.

- **directory**: String. The path to a directory for storing the search outputs. Defaults to None, which would create a folder with the name of the AutoModel in the current directory.
- **objective**: String. Name of model metric to minimize or maximize, e.g. 'val_accuracy'. Defaults to 'val_loss'.
- **seed**: Int. Random seed.

Coming Soon:

StructuredDataClassifier

StructuredDataRegressor

TimeSeriesForecaster

New
Home
Author
Notebooks
Automated ML
Designer
Assets
Datasets
Experiments
Pipelines
Models
Endpoints
Manage
Compute
Datastores
Data labeling

automl_tianic

Switch to old experience

Edit table | Refresh | Reset to default view | Include child runs

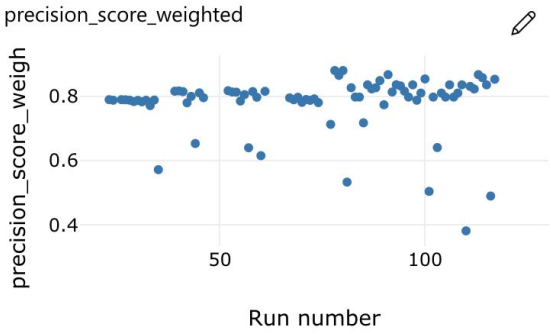
Run status

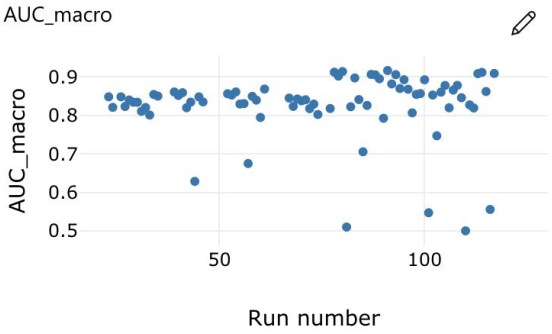
4 Running

99 Completed

12 Failed

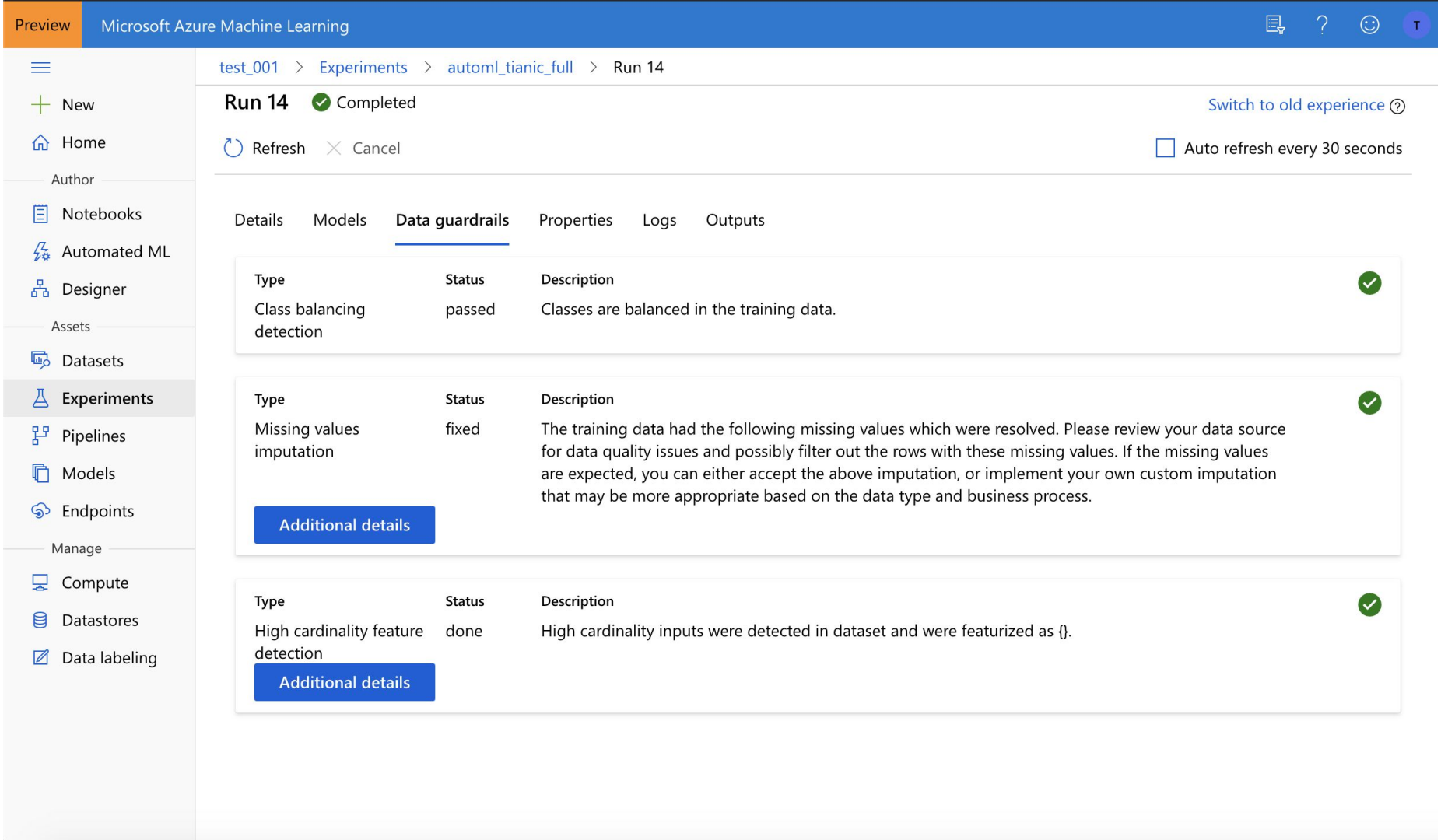
7 Other

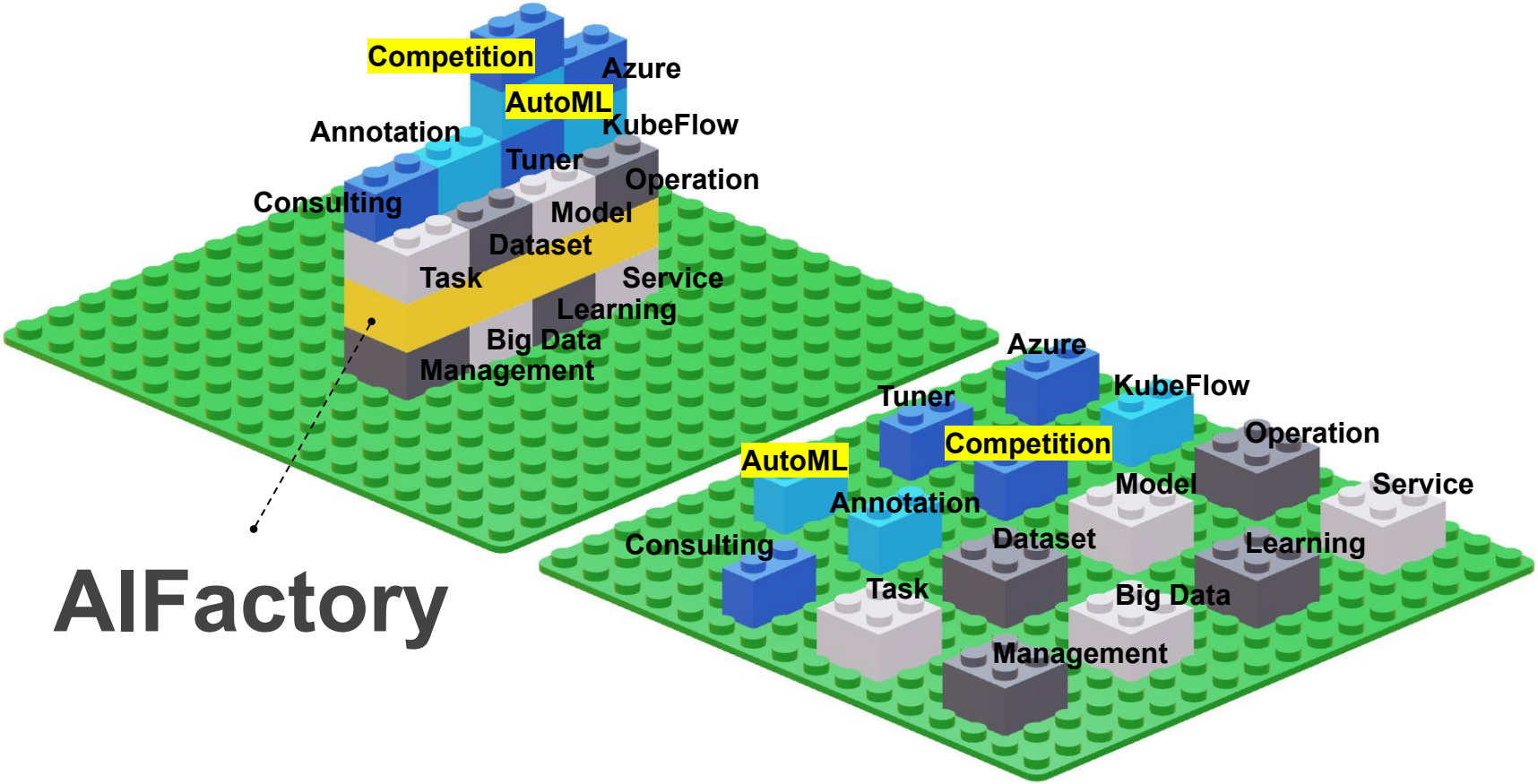




Add filter

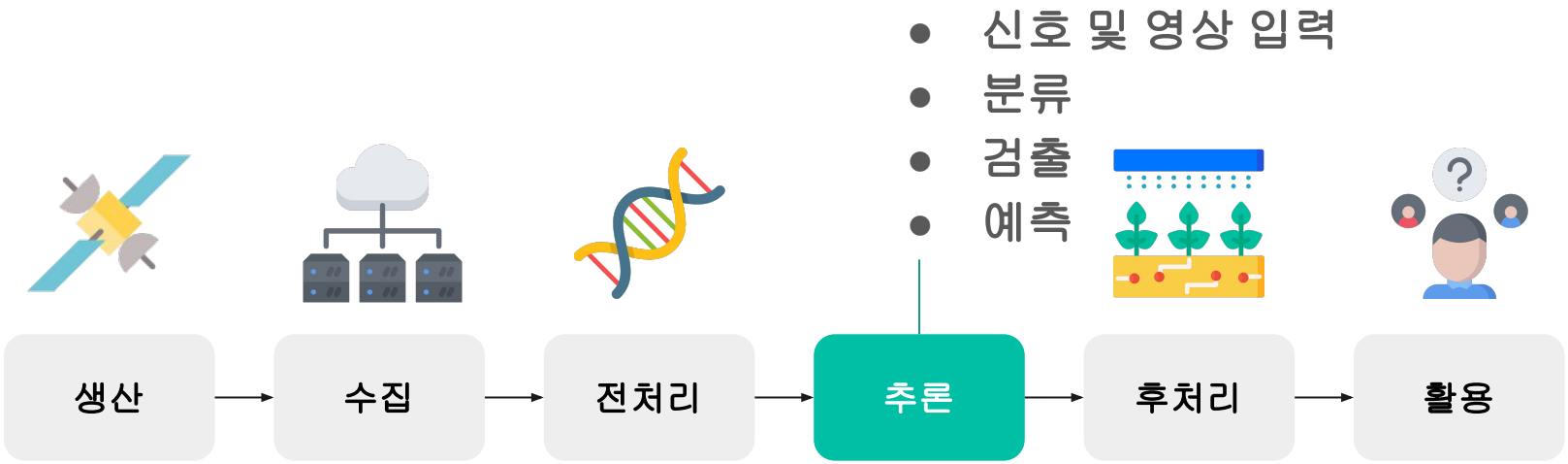
Run	Created time	Duration	Status	Compute target	Run type	precision_s...	AUC_macro	Tags
Run 123	December 14, 2019 5:2...	1m 11s	Completed	testv101	automl.model_exp...			model_explain
Run 122	December 14, 2019 5:2...	1m 6s	Completed	testv101	automl.model_exp...			model_explain
Run 121	December 14, 2019 5:0...	2m 45s	Completed	testv101	automl.model_exp...			model_explain
Run 120	December 14, 2019 4:5...	23s	Canceled	testv101	azureml.scriptrun			model_explain


Preview
Microsoft Azure Machine Learning
T
New
Home
Author
Notebooks
Automated ML
Designer
Assets
Datasets
Experiments
Pipelines
Models
Endpoints
Manage
Compute
Datastores
Data labeling
test_001
Experiments
automl_tianic_full
Run 14
Run 14
Completed
Switch to old experience
Refresh
Cancel
Auto refresh every 30 seconds
Details
Models
Data guardrails
Properties
Logs
Outputs
Type
Status
Description
Class balancing detection
passed
Classes are balanced in the training data.
Type
Status
Description
Missing values imputation
fixed
The training data had the following missing values which were resolved. Please review your data source for data quality issues and possibly filter out the rows with these missing values. If the missing values are expected, you can either accept the above imputation, or implement your own custom imputation that may be more appropriate based on the data type and business process.
Additional details
Type
Status
Description
High cardinality feature detection
done
High cardinality inputs were detected in dataset and were featurized as {}.
Additional details


Competition
Azure
AutoML
KubeFlow
Annotation
Tuner
Operation
Consulting
Model
Dataset
Task
Service
Learning
Big Data
Management
Azure
KubeFlow
Tuner
Operation
Competition
AutoML
Model
Service
Annotation
Consulting
Dataset
Learning
Task
Big Data
Management

AIFactory

비지니스 모델 : 데이터셋 구축



- 데이터는 중요하나 프로세스마다 그 가치가 다름
- 인공지능 추론의 기술 가치는 하락하고 있음
- 일부 데이터만 공개, 비식별 조치 >> 공개 조건 달성
- 데이터셋 암호화 기술 발전 >> 폐쇄 학습 가능

TF Encrypted Keras

- Encrypted Training
- Encrypted Predictions with Public Training
- Encrypted Predictions After Public Training with Differential Privacy

Encrypted Training

Using TFE Keras, Data Scientists can train models on encrypted data

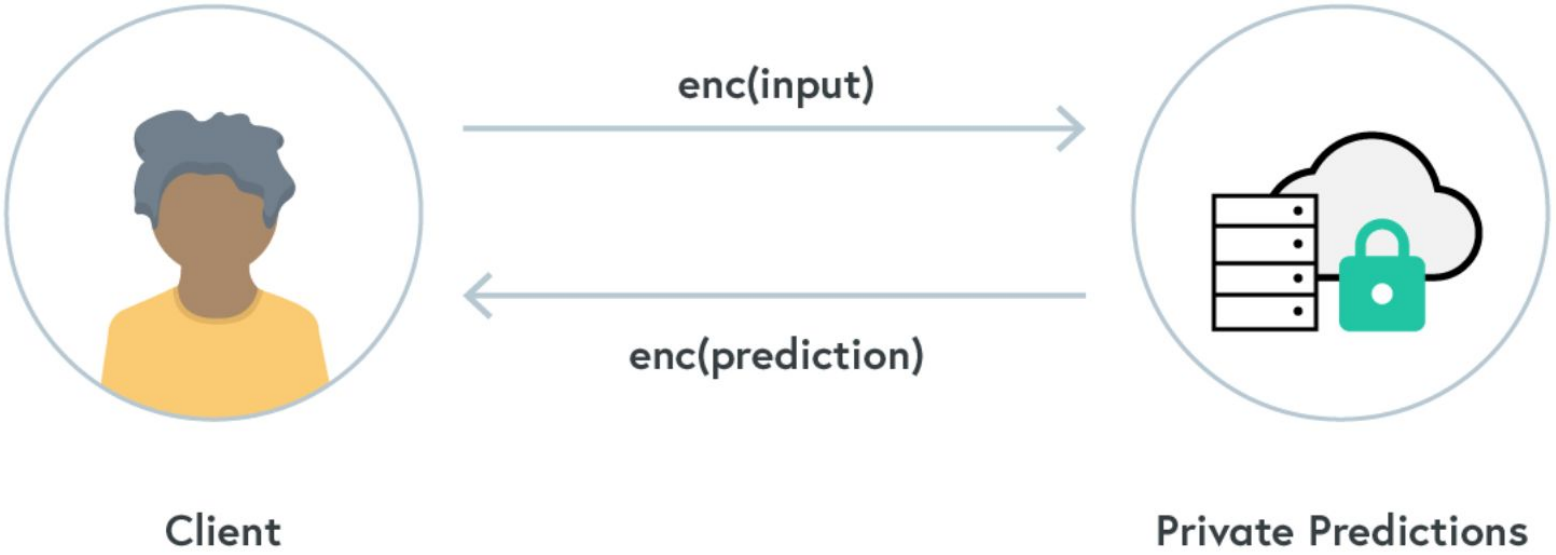
Ref. https://medium.com/dropoutlabs/encrypted-deep-learning-training-and-predictions-with-tf-encrypted-keras-557193284f44

Encrypted Training

```
import tf_encrypted as tfe
from tf_encrypted.keras.losses import BinaryCrossentropy
from tf_encrypted.keras.optimizers import SGD

from common import DataOwner

num_features = 10
batch_size = 100
steps_per_epoch = (training_set_size // batch_size)
epochs = 20

# Provide encrypted training data
data_owner = DataOwner('data-owner', batch_size)
x_train, y_train = data_owner.provide_private_training_data()

# Define model
model = tfe.keras.Sequential()
model.add(tfe.keras.layers.Dense(1, batch_input_shape=[batch_size, nu
model.add(tfe.keras.layers.Activation('sigmoid'))

# Specify optimizer and loss
model.compile(optimizer=SGD(lr=0.01),
              loss=BinaryCrossentropy())

# Start training
model.fit(x_train,
          y_train,
          epochs=epochs,
          steps_per_epoch=steps_per_epoch)
```

Ref. https://medium.com/dropoutlabs/encrypted-deep-learning-training-and-predictions-with-tf-encrypted-keras-557193284f44

Encrypted Predictions with Public Training



Using TFE Keras, users only share encrypted data to get the prediction

Ref. https://medium.com/dropoutlabs/encrypted-deep-learning-training-and-predictions-with-tf-encrypted-keras-557193284f44

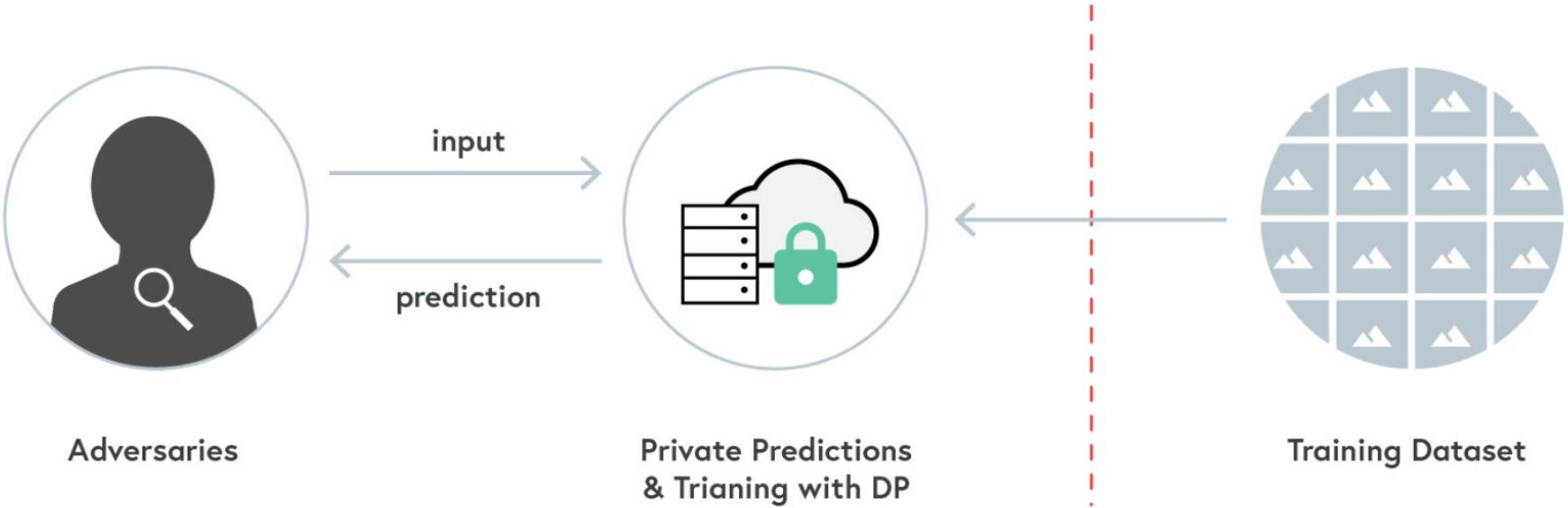
Encrypted Predictions with Public Training

```
import tensorflow as tf
import tf_encrypted as tfe

# Define plaintext model with tf Keras
model = tf.keras.Sequential([
        tf.keras.layers.Conv2D(16, 8,
                               strides=2,
                               padding='same',
                               activation='relu',
                               batch_input_shape=input_shape),
        tf.keras.layers.AveragePooling2D(2, 1),
        tf.keras.layers.Conv2D(32, 4,
                               strides=2,
                               padding='valid',
                               activation='relu'),
        tf.keras.layers.AveragePooling2D(2, 1),
        tf.keras.layers.Flatten(),
        tf.keras.layers.Dense(32, activation='relu'),
        tf.keras.layers.Dense(10, name='logit')
  ])

# load trained weights
pre_trained_weights = 'short-dnn.h5'
model.load_weights(pre_trained_weights)

# create private model from tf Keras model
tfe_model = tfe.keras.models.clone_model(model)
```

Ref. https://medium.com/dropoutlabs/encrypted-deep-learning-training-and-predictions-with-tf-encrypted-keras-557193284f44

Encrypted Predictions After Public Training with Differential Privacy

- membership inference
- model inversion.



Differential Privacy prevents the model from memorizing sensitive data

Ref. https://medium.com/dropoutlabs/encrypted-deep-learning-training-and-predictions-with-tf-encrypted-keras-557193284f44

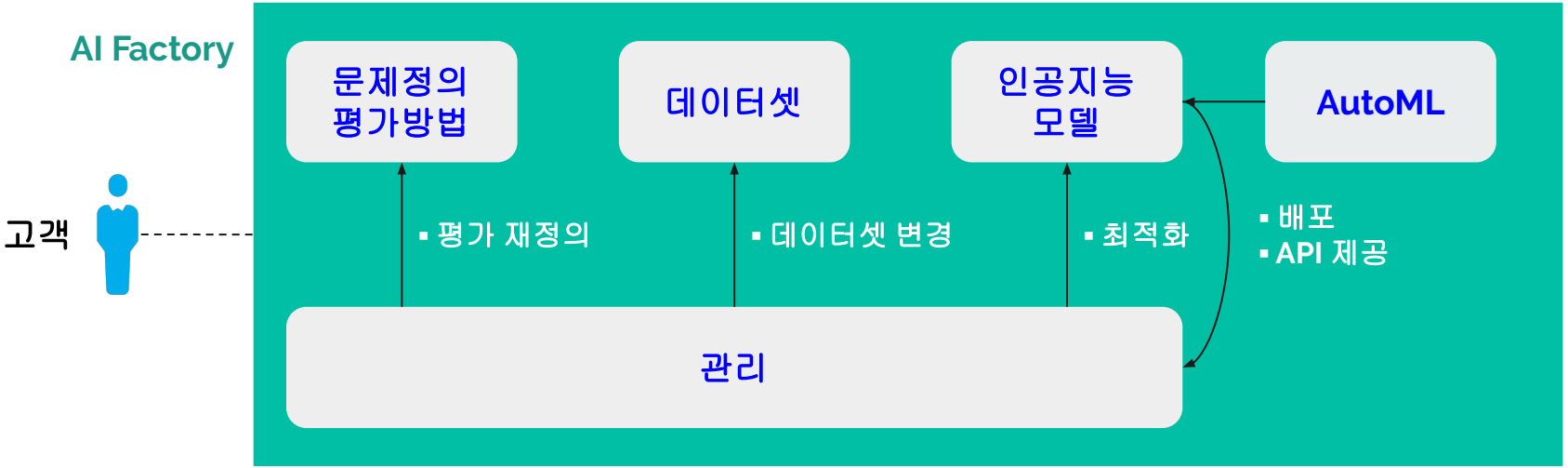
Encrypted Predictions After Public Training with Differential Privacy

```
if FLAGS.dpsgd:
  ledger = privacy_ledger.PrivacyLedger(
      population_size=60000,
      selection_probability=(FLAGS.batch_size / 60000))

  # Use DP version of GradientDescentOptimizer. Other optimizers are
  # available in dp_optimizer. Most optimizers inheriting from
  # tf.train.Optimizer should be wrappable in differentially private
  # counterparts by calling dp_optimizer.optimizer_from_args().
  optimizer = dp_optimizer.DPGradientDescentGaussianOptimizer(
      l2_norm_clip=FLAGS.l2_norm_clip,
      noise_multiplier=FLAGS.noise_multiplier,
      num_microbatches=FLAGS.microbatches,
      ledger=ledger,
      learning_rate=FLAGS.learning_rate)
  training_hooks = [
      EpsilonPrintingTrainingHook(ledger)
  ]
  opt_loss = vector_loss
```

```
else:
  optimizer = GradientDescentOptimizer(learning_rate=FLAGS.learning_rate)
  training_hooks = []
  opt_loss = scalar_loss
global_step = tf.compat.v1.train.get_global_step()
train_op = optimizer.minimize(loss=opt_loss, global_step=global_step)
# In the following, we pass the mean of the loss (scalar_loss) rather than
# the vector_loss because tf.estimator requires a scalar loss. This is only
# used for evaluation and debugging by tf.estimator. The actual loss being
# minimized is opt_loss defined above and passed to optimizer.minimize().
return tf.estimator.EstimatorSpec(mode=mode,
                                  loss=scalar_loss,
                                  train_op=train_op,
                                  training_hooks=training_hooks)
```

Ref. https://medium.com/dropoutlabs/encrypted-deep-learning-training-and-predictions-with-tf-encrypted-keras-557193284f44

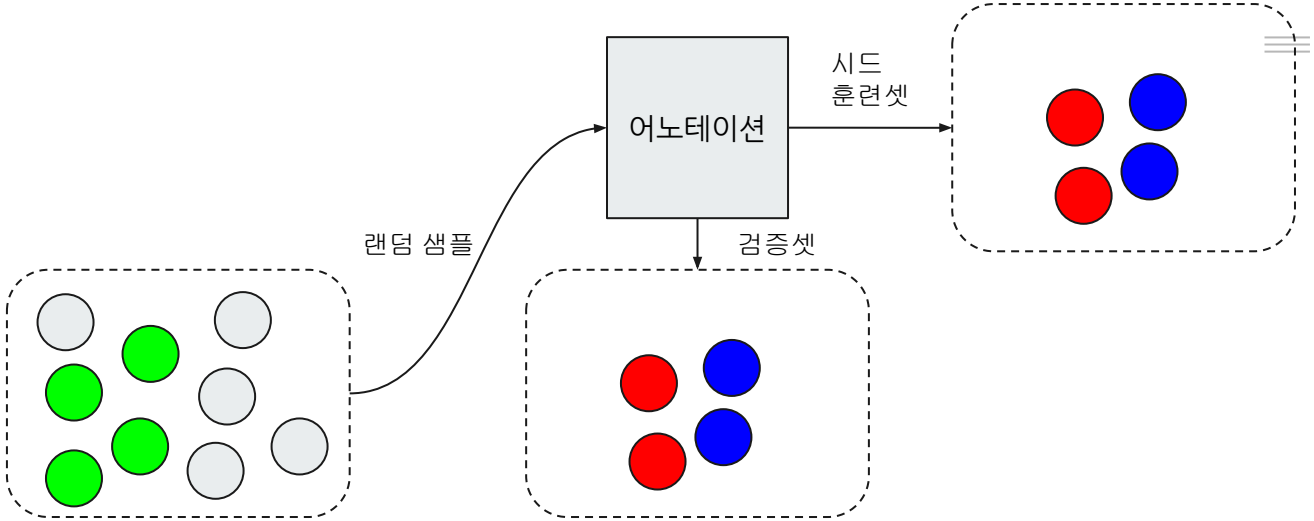
인공지능 모델 유지 관리 비용은 만만치 않다 >> 자동 관리

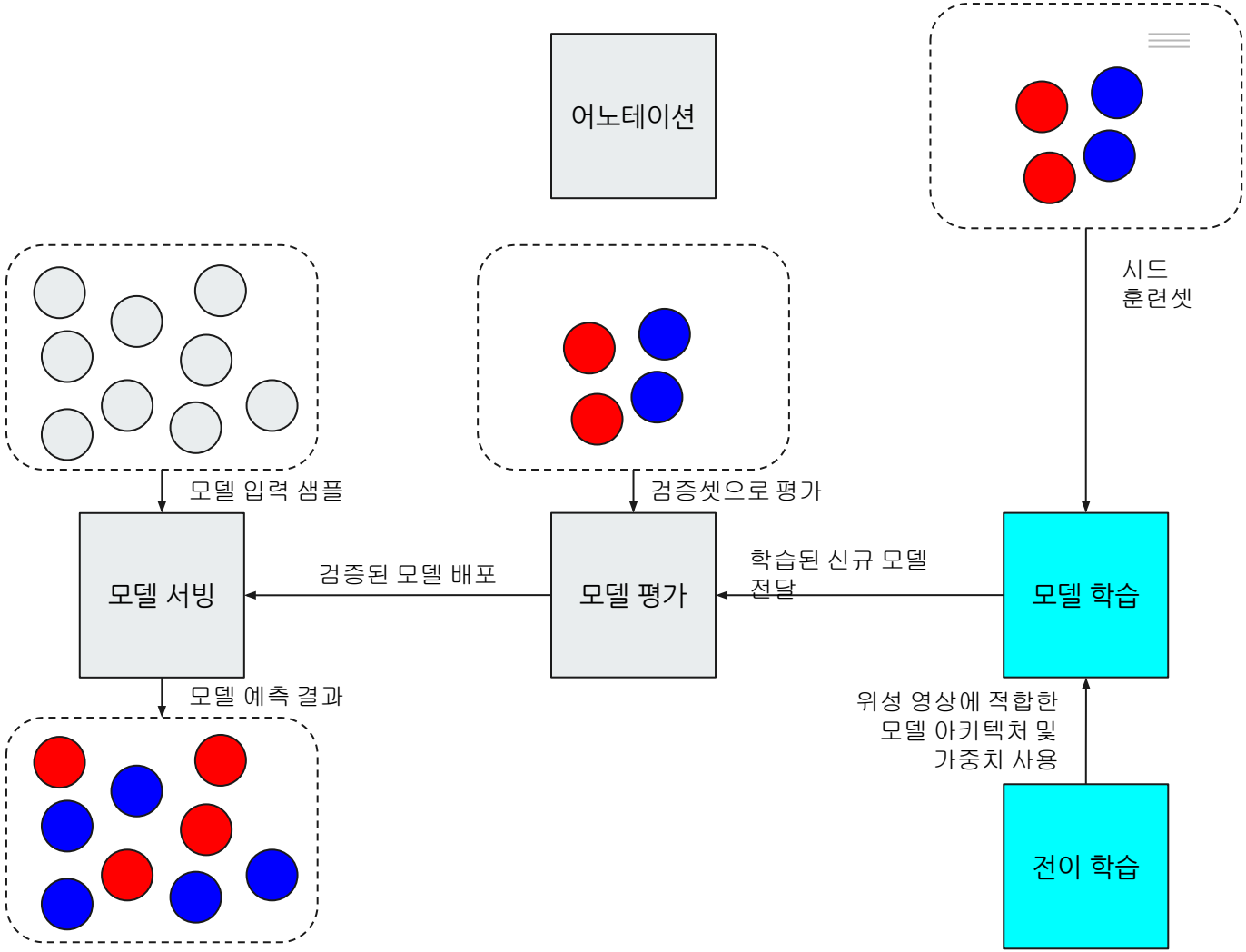


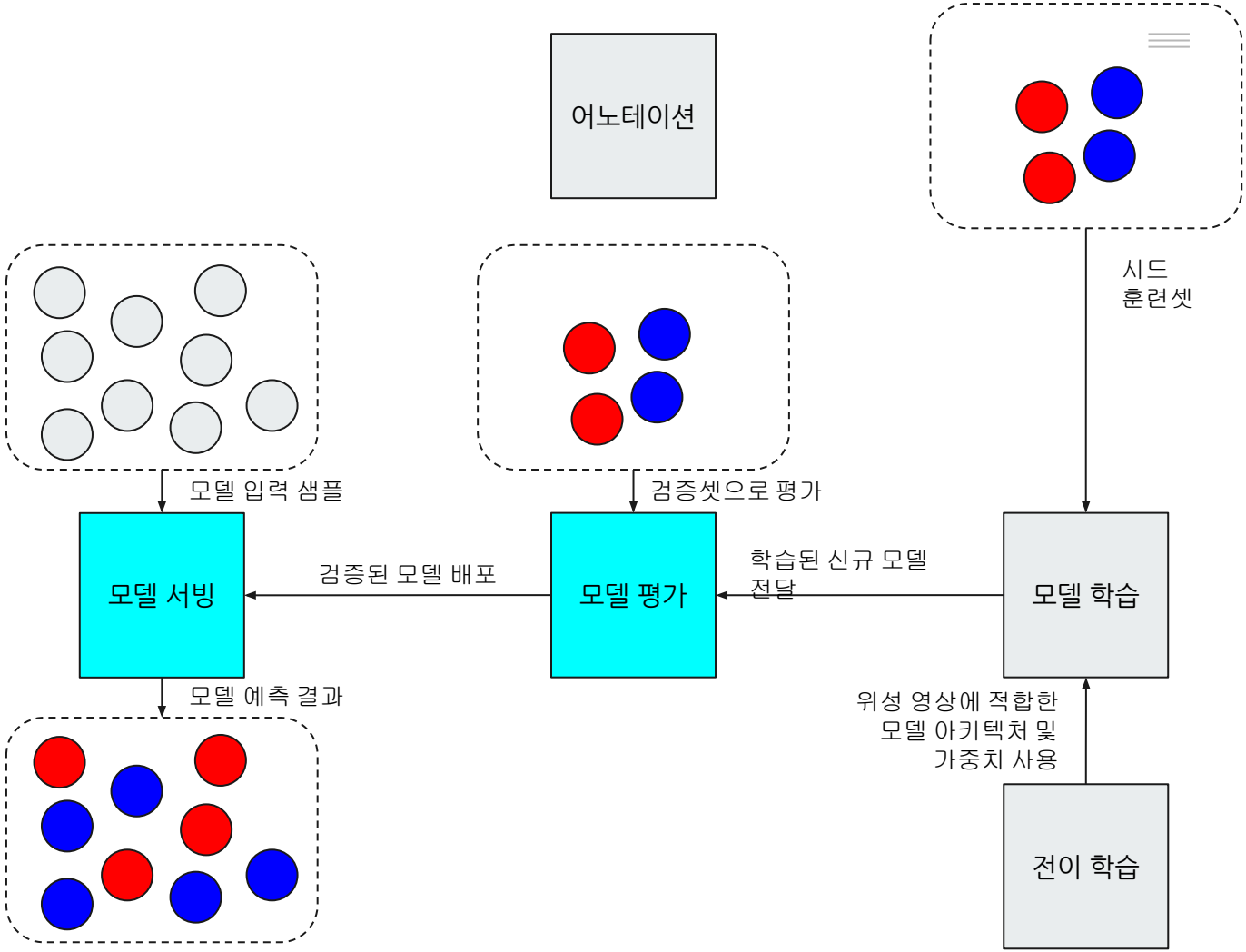
- 세 요소 중 하나가 변경 >> 자동으로 성능 업데이트
- 액티브/온라인 러닝 지원 >> 데이터셋 효율 생산 및 변경 관리
- 자동 하이퍼파라미터 튜닝 >> 모델 최적화
- 모델 배포 및 Open API 제공 >> 손쉬운 서비스 연계

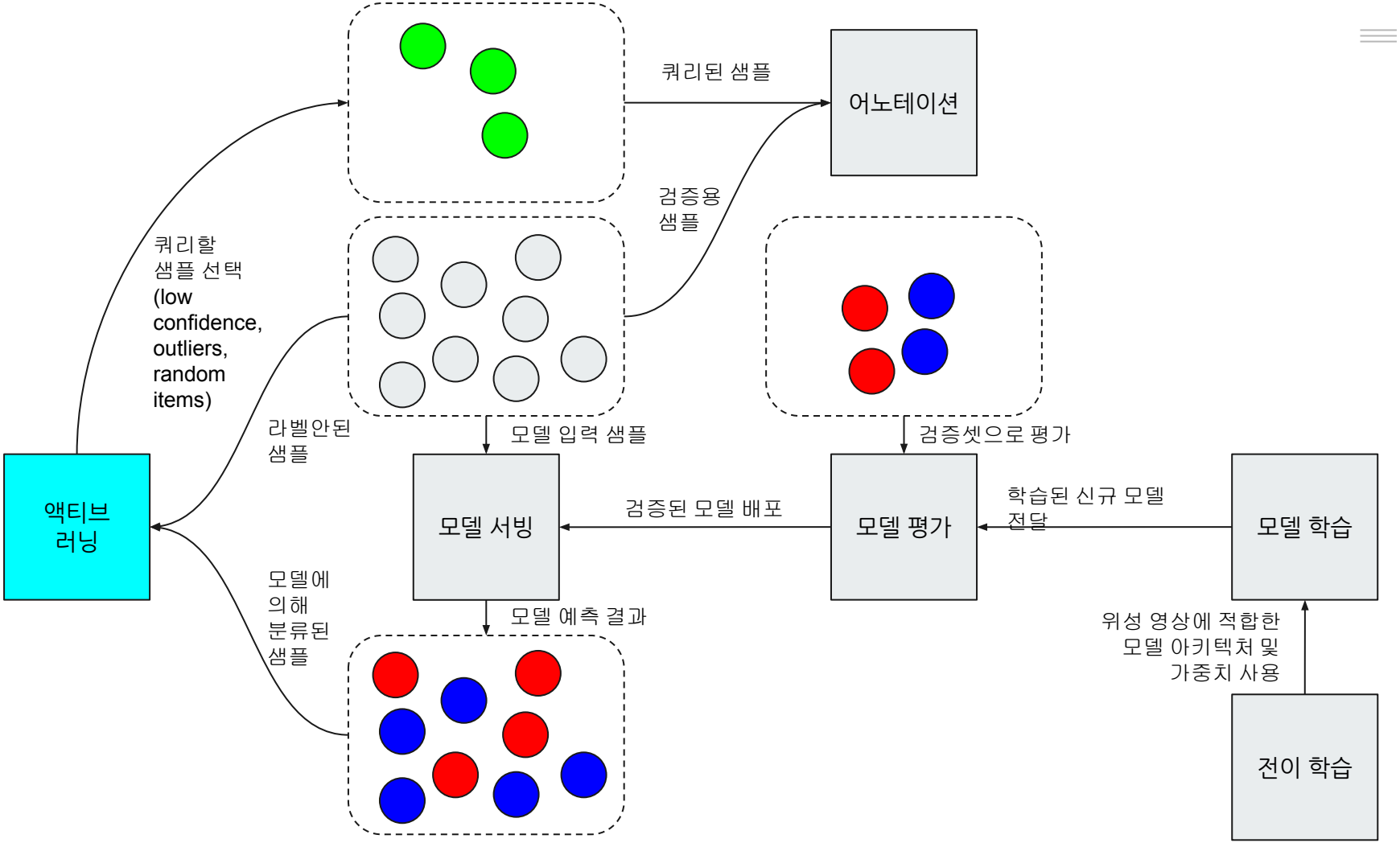
Active Learning

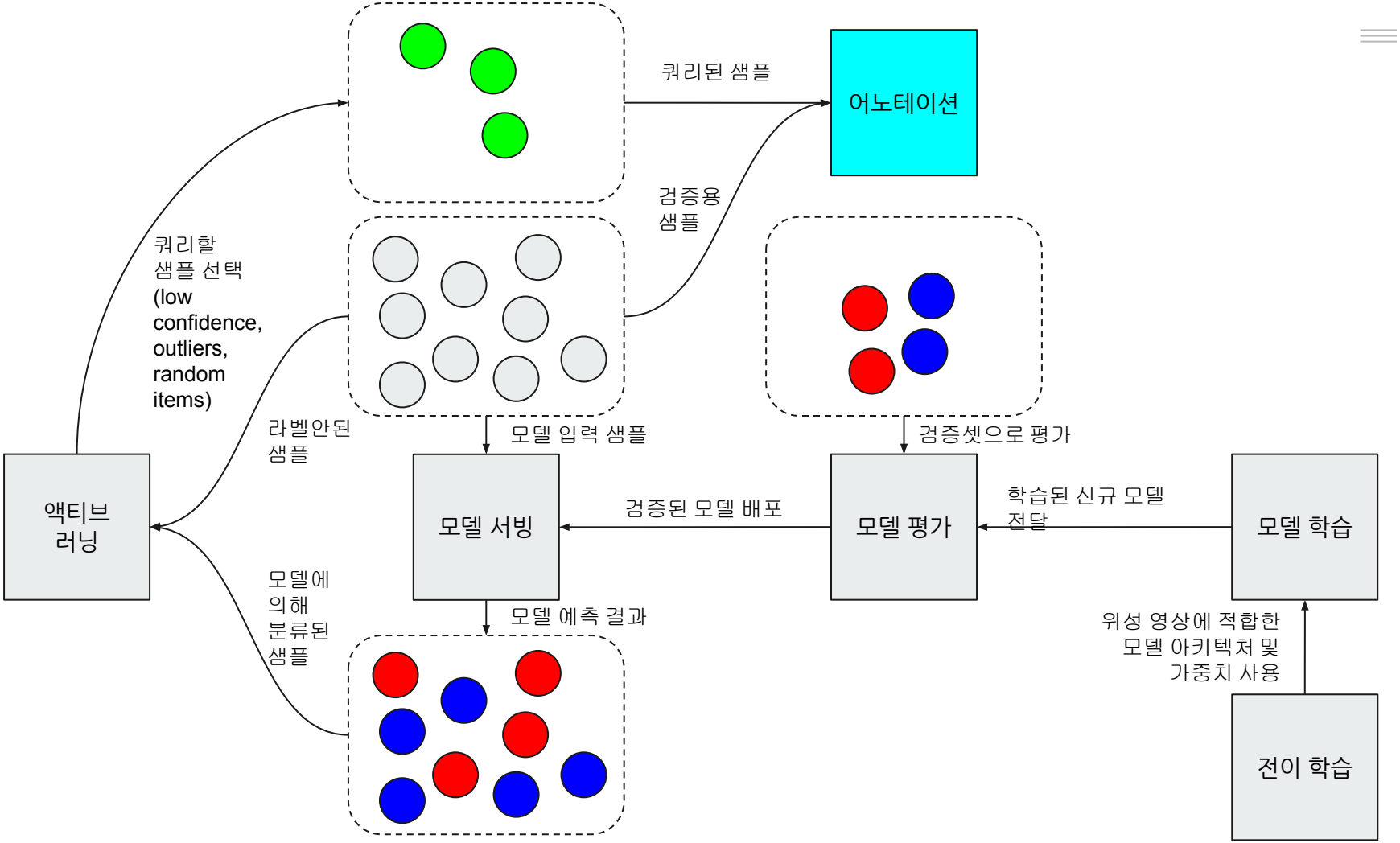
- 문제점
  - 데이터셋 양이 방대하여 모두 라벨링을 하기에 제한이 있을 때
- 해결방안
  - 라벨링 돼있지 않은 데이터셋에 한하여 보다 효율적인 데이터에 한해 전문가에게 라벨링 요청을 함
    - 효율적인 데이터를 판단하는 기준
    - https://inspaceai.github.io/2019/06/05/ActiveLearning_Introduction/
      - Cross-Entropy
      - Least Confidence
      - Margin Sampling

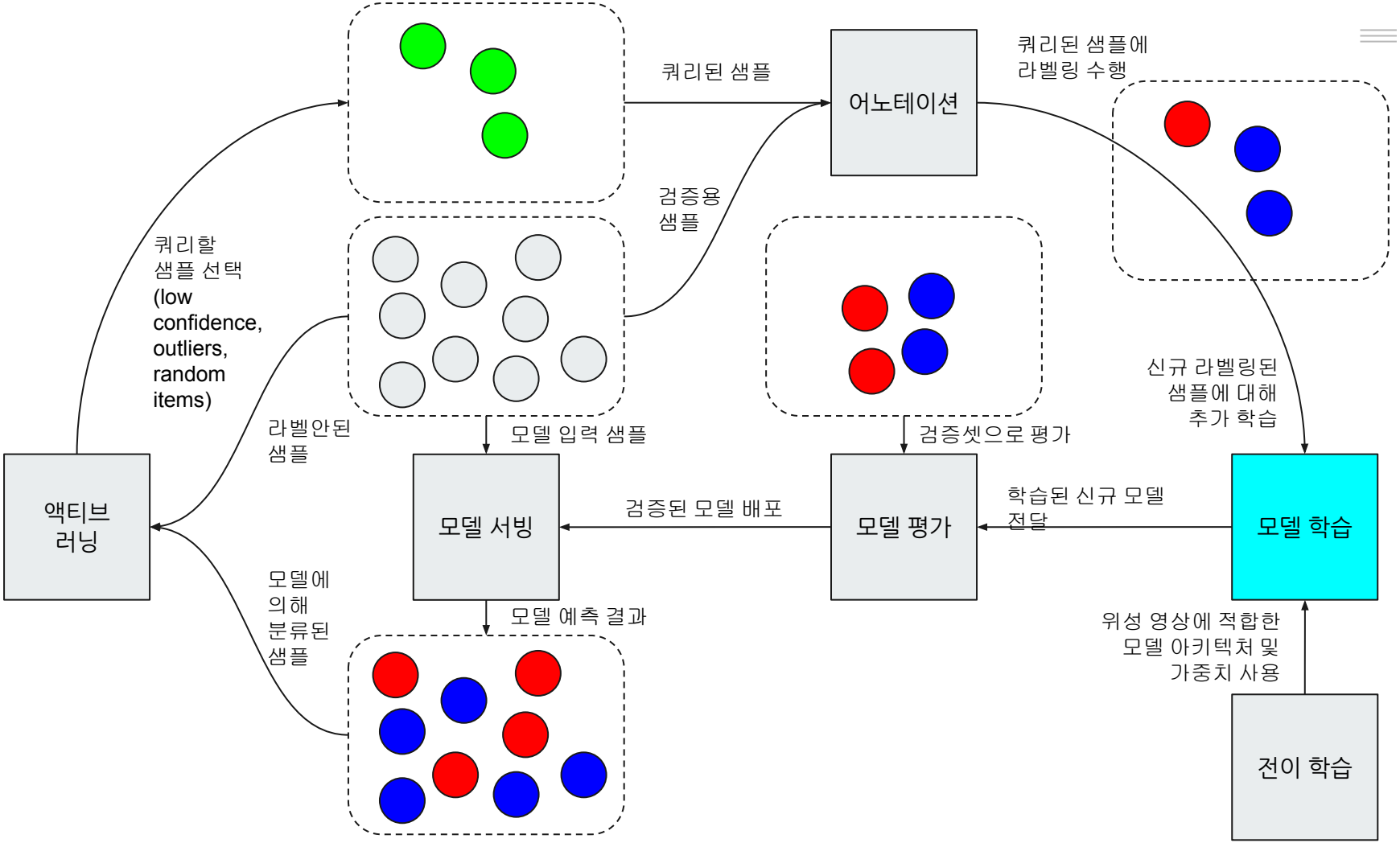

어노테이션
시드
훈련셋
랜덤 샘플
검증셋

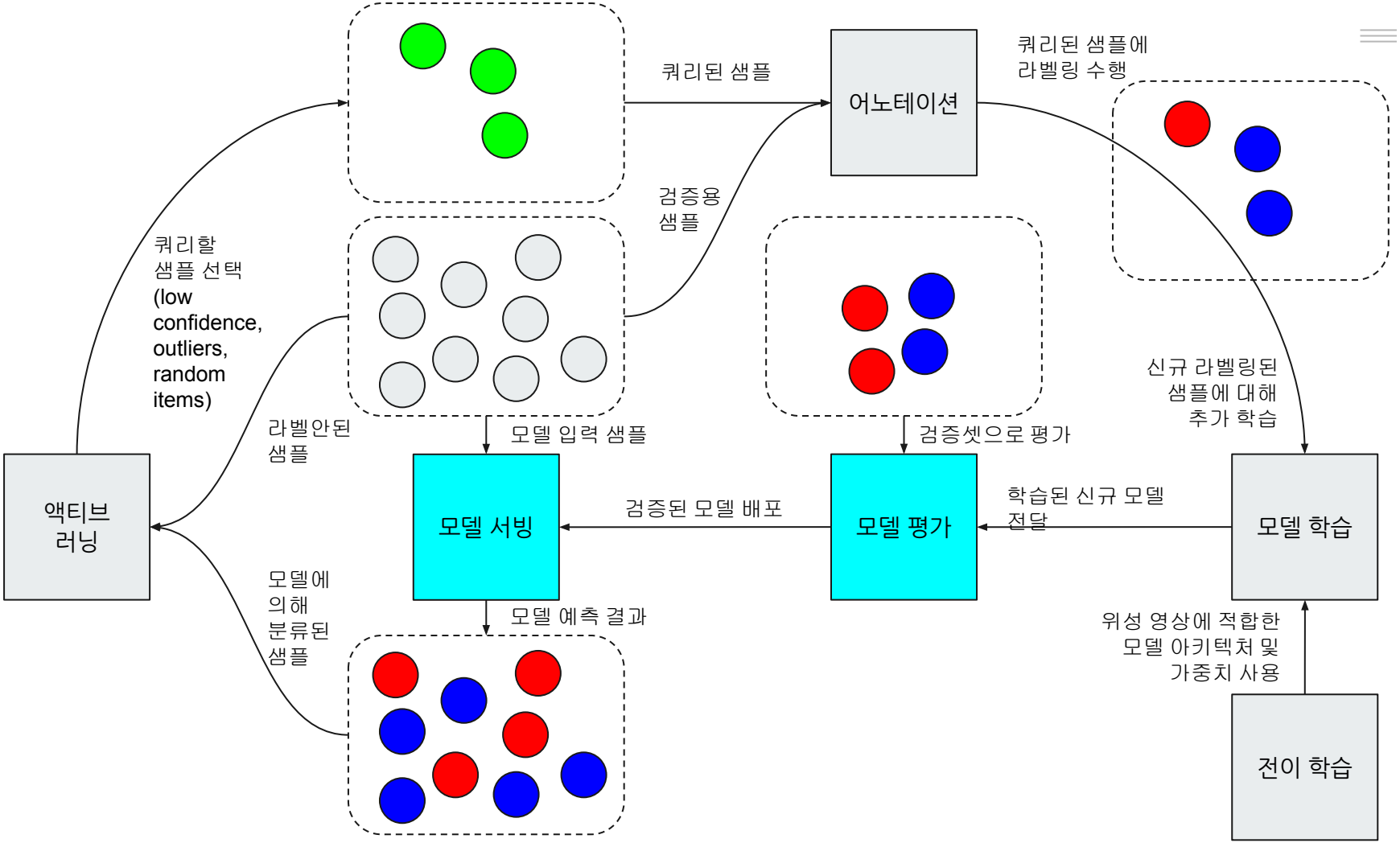

어노테이션
시드
훈련셋
모델 입력 샘플
검증셋으로 평가
모델 서빙
검증된 모델 배포
모델 평가
학습된 신규 모델
전달
모델 학습
모델 예측 결과
위성 영상에 적합한
모델 아키텍처 및
가중치 사용
전이 학습

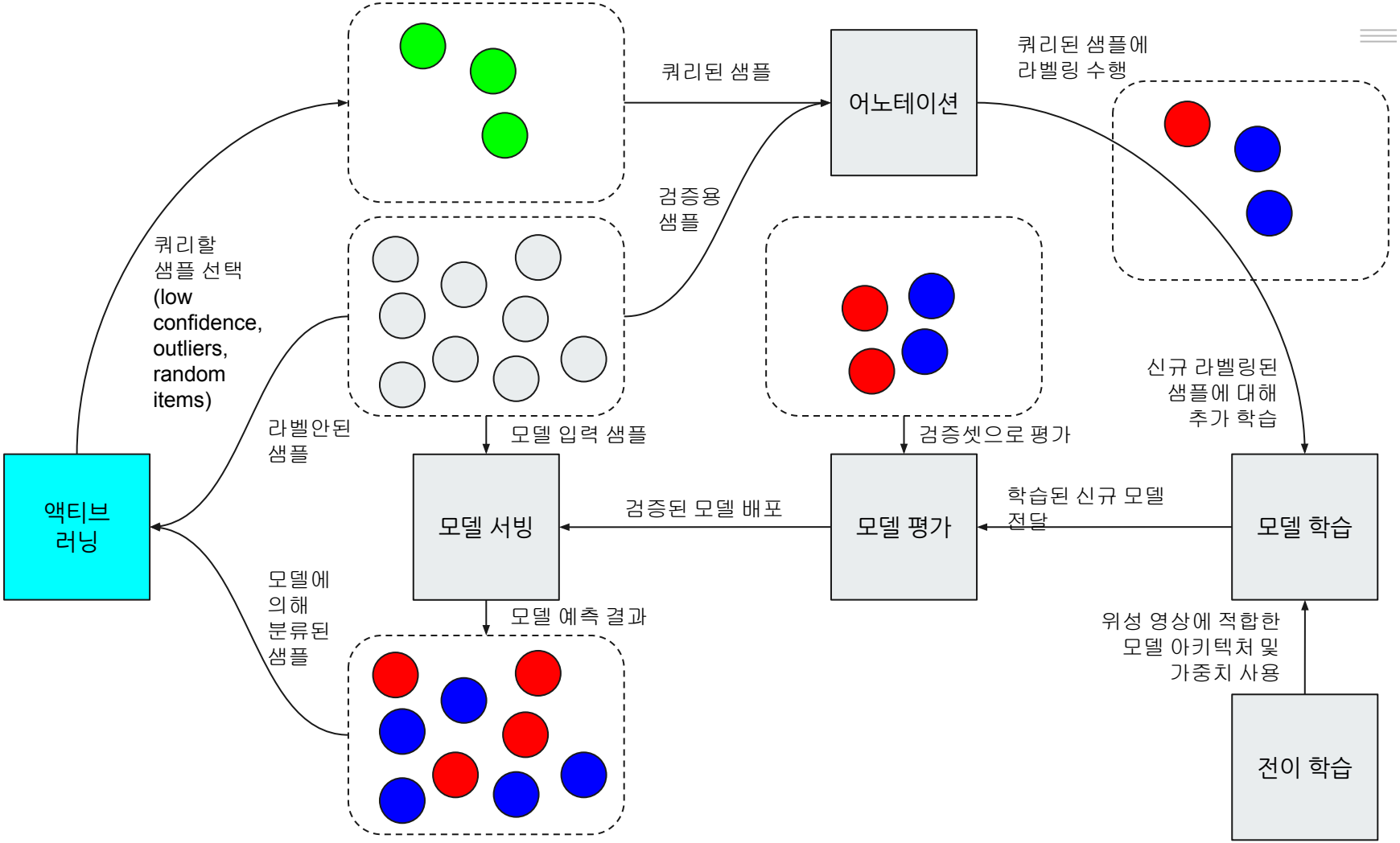

어노테이션
시드
훈련셋
모델 입력 샘플
검증셋으로 평가
모델 서빙
검증된 모델 배포
모델 평가
학습된 신규 모델
전달
모델 학습
모델 예측 결과
위성 영상에 적합한
모델 아키텍처 및
가중치 사용
전이 학습

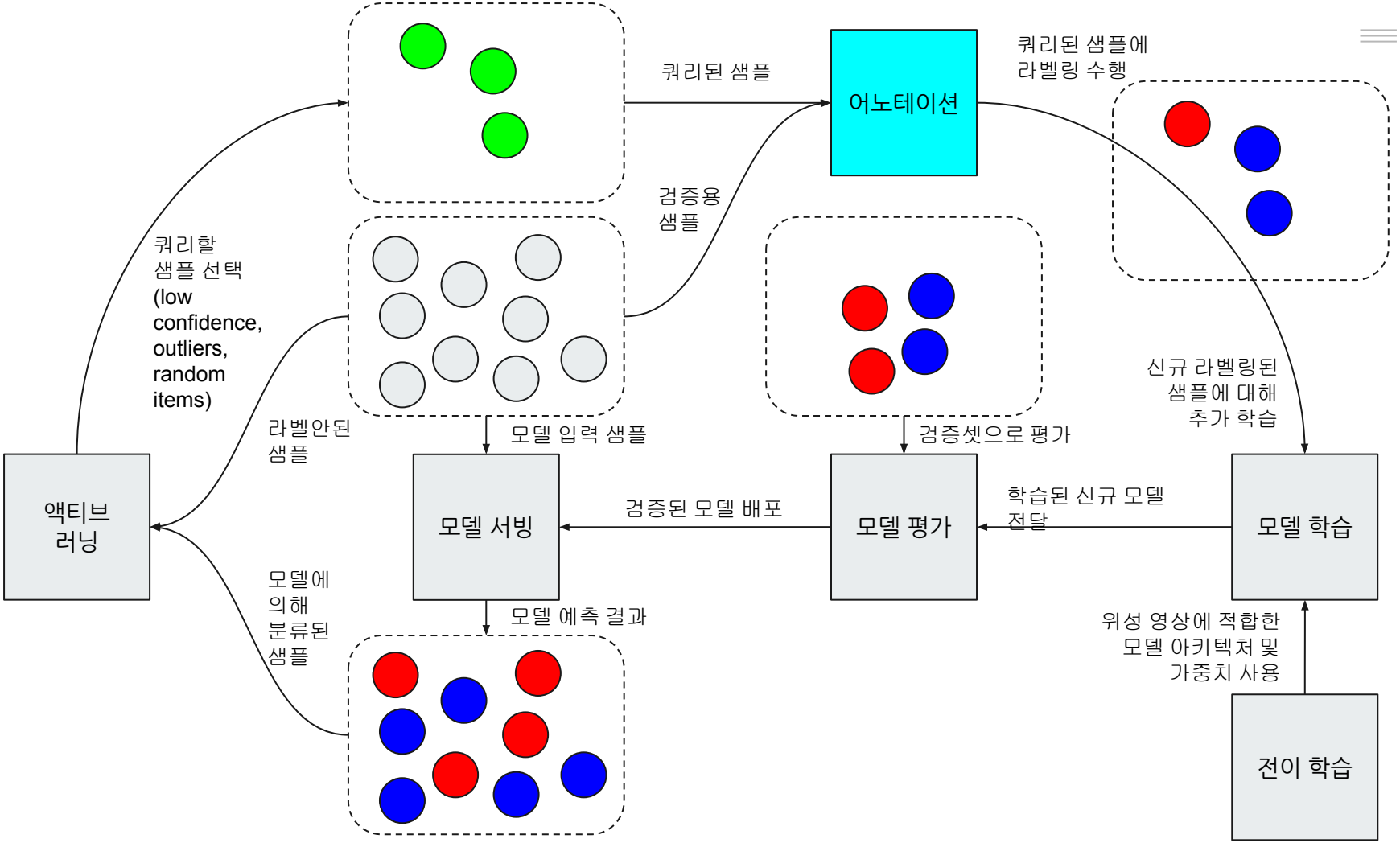

쿼리된 샘플
어노테이션
검증용
샘플
쿼리할
샘플 선택
(low
confidence,
outliers,
random
items)
라벨안된
샘플
모델 입력 샘플
검증셋으로 평가
액티브
러닝
모델 서빙
검증된 모델 배포
모델 평가
학습된 신규 모델
전달
모델 학습
모델에
의해
분류된
샘플
모델 예측 결과
위성 영상에 적합한
모델 아키텍처 및
가중치 사용
전이 학습


쿼리된 샘플
어노테이션
검증용
샘플
쿼리할
샘플 선택
(low
confidence,
outliers,
random
items)
라벨안된
샘플
모델 입력 샘플
검증셋으로 평가
액티브
러닝
모델 서빙
검증된 모델 배포
모델 평가
학습된 신규 모델
전달
모델 학습
모델에
의해
분류된
샘플
모델 예측 결과
위성 영상에 적합한
모델 아키텍처 및
가중치 사용
전이 학습


쿼리된 샘플
어노테이션
쿼리된 샘플에
라벨링 수행
검증용
샘플
쿼리할
샘플 선택
(low
confidence,
outliers,
random
items)
라벨안된
샘플
모델 입력 샘플
검증셋으로 평가
신규 라벨링된
샘플에 대해
추가 학습
액티브
러닝
모델 서빙
검증된 모델 배포
모델 평가
학습된 신규 모델
전달
모델 학습
모델에
의해
분류된
샘플
모델 예측 결과
위성 영상에 적합한
모델 아키텍처 및
가중치 사용
전이 학습

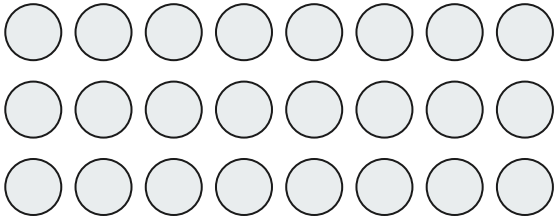

쿼리된 샘플
어노테이션
쿼리된 샘플에
라벨링 수행
검증용
샘플
쿼리할
샘플 선택
(low
confidence,
outliers,
random
items)
라벨안된
샘플
모델 입력 샘플
검증셋으로 평가
신규 라벨링된
샘플에 대해
추가 학습
액티브
러닝
모델 서빙
검증된 모델 배포
모델 평가
학습된 신규 모델
전달
모델 학습
모델에
의해
분류된
샘플
모델 예측 결과
위성 영상에 적합한
모델 아키텍처 및
가중치 사용
전이 학습


액티브 러닝
쿼리할 샘플 선택 (low confidence, outliers, random items)
쿼리된 샘플
어노테이션
쿼리된 샘플에 라벨링 수행
검증용 샘플
라벨안된 샘플
모델 입력 샘플
검증셋으로 평가
신규 라벨링된 샘플에 대해 추가 학습
모델 서빙
검증된 모델 배포
모델 평가
학습된 신규 모델 전달
모델 학습
모델에 의해 분류된 샘플
모델 예측 결과
위성 영상에 적합한 모델 아키텍처 및 가중치 사용
전이 학습

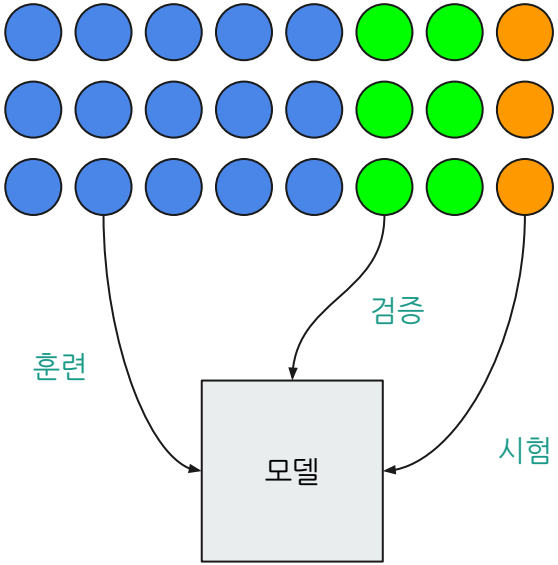

쿼리된 샘플
어노테이션
쿼리된 샘플에
라벨링 수행
검증용
샘플
쿼리할
샘플 선택
(low
confidence,
outliers,
random
items)
라벨안된
샘플
모델 입력 샘플
검증셋으로 평가
신규 라벨링된
샘플에 대해
추가 학습
액티브
러닝
모델 서빙
검증된 모델 배포
모델 평가
학습된 신규 모델
전달
모델 학습
모델에
의해
분류된
샘플
모델 예측 결과
위성 영상에 적합한
모델 아키텍처 및
가중치 사용
전이 학습

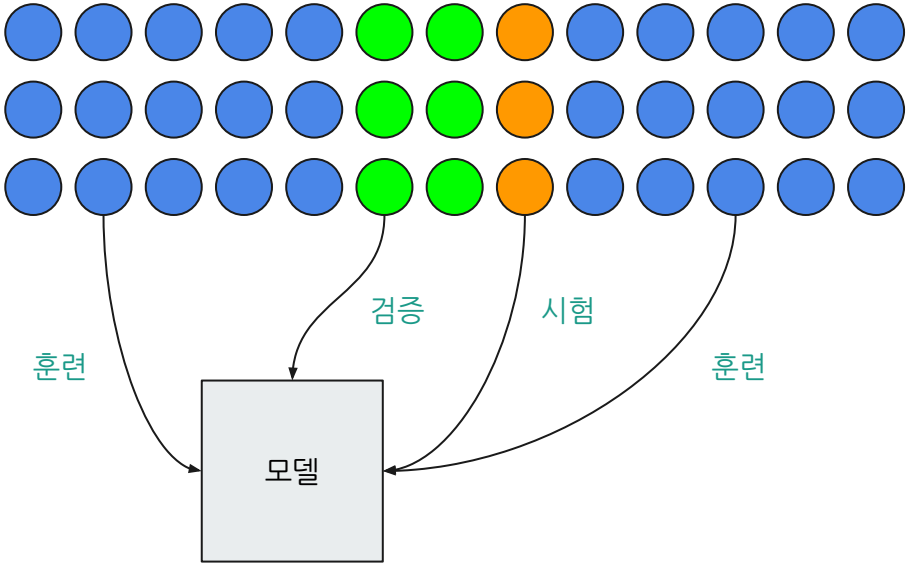
Online Learning

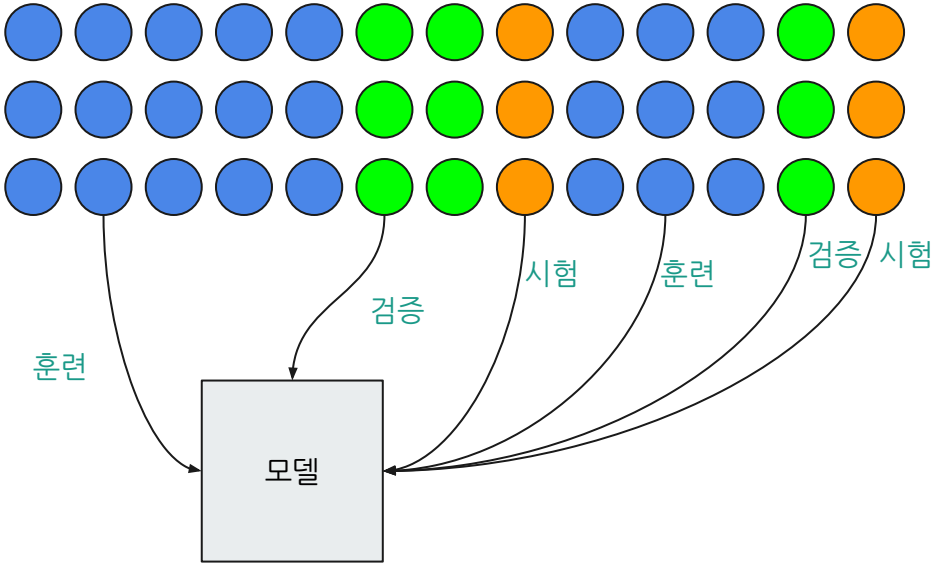
- 유사 용어
  - Incremental Learning, Streaming Learning
- 문제점
  - 데이터 셋(라벨링 포함)이 너무 방대하여 하드웨어 자원이 부족할 때
  - 데이터 셋(라벨링 포함)이 스트리밍 형식으로 매 시간 생성될 때
- 해결방안
  - 메모리에 적재할 수 있는 만큼 가져와 배치학습 실시
  - 데이터 셋이 생성되는 대로 배치학습 실시

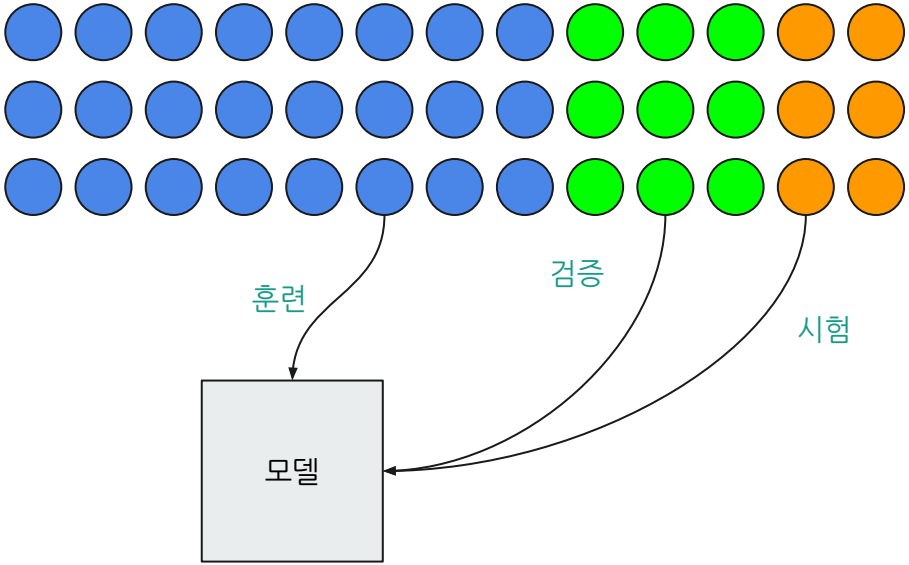


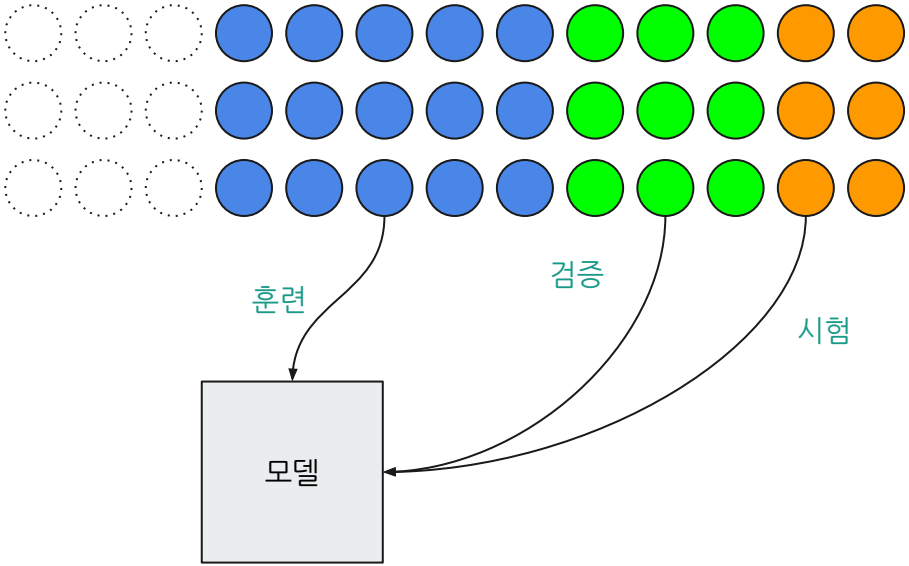
모델


훈련
검증
시험
모델

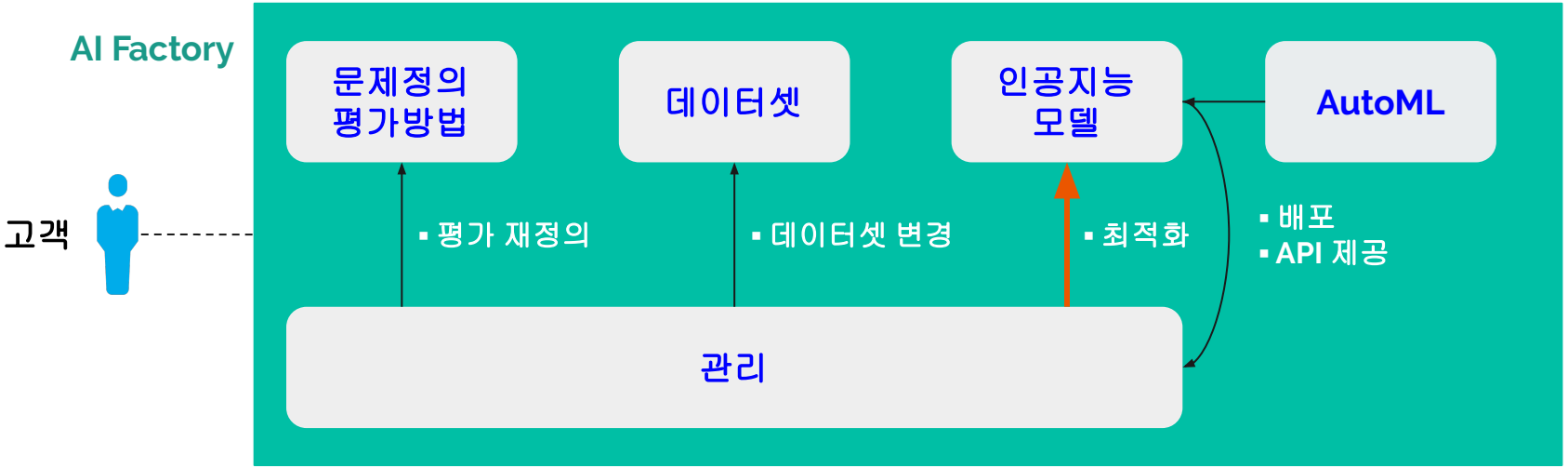

훈련
검증
시험
훈련
모델


훈련
검증
시험
훈련
검증
시험
모델


훈련
검증
시험
모델


훈련
검증
시험
모델

인공지능 모델 유지 관리 비용은 만만치 않다 >> 자동 관리



- 세 요소 중 하나가 변경 >> 자동으로 성능 업데이트
- 액티브/온라인 러닝 지원 >> 데이터셋 효율 생산 및 변경 관리
- 자동 하이퍼파라미터 튜닝 >> 모델 최적화
- 모델 배포 및 Open API 제공 >> 손쉬운 서비스 연계

Hyperparameter Tuning >> Keras Tuner

Keras Tuner documentation

Installation

Requirements:

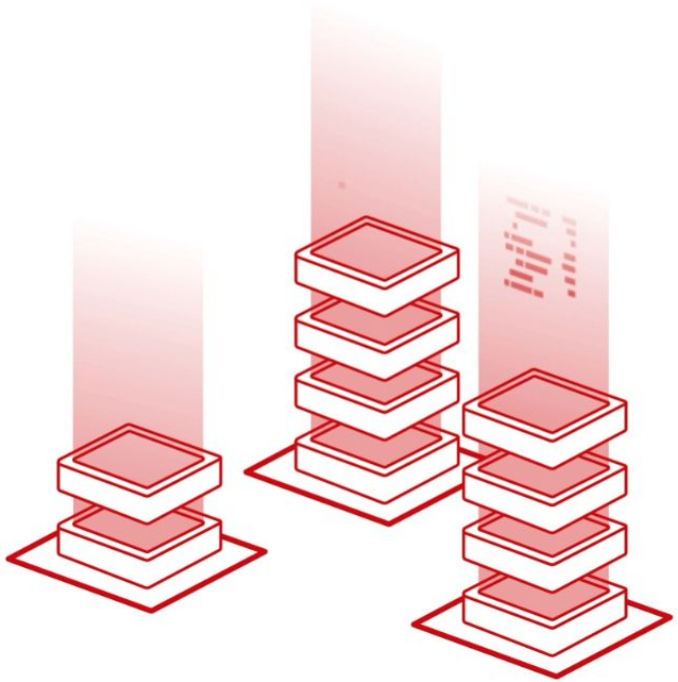
- Python 3.6
- TensorFlow 2.0

Install latest release:

```
pip install -U keras-tuner
```

Install from source:

```
git clone https://github.com/keras-team/keras-tuner.git
```



Getting the optimal model requires to tune many inter-dependent parameters

```
model = Sequential()
model.add(Conv2D(32, kernel_size=(3, 3), activation='relu',
   input_shape=(28, 28, 1)))
model.add(Conv2D(64, kernel_size=(3, 3), activation='relu'))
model.add(Flatten())
model.add(Dense(20, activation='relu'))
model.add(Dropout(0.2))
model.add(Dense(10, activation='softmax'))
model.compile(loss='categorical_crossentropy',
   optimizer=Adam(0.001))
model.summary()
```

- 첫번째 Conv2D의 필터 수 : L1_NUM_FILTERS
- 두번째 Conv2D의 필터 수 : L2_NUM_FILTERS
- Dense - Dropout 은닉층의 반복횟수 : NUM_LAYERS
- Dense 레이어의 출력뉴런 수 : NUM_DIMS
- Dropout 레이어의 드랍율 : DROPOUT_RATE
- 최적화기의 학습률 : LR

```
LR = Choice('learning_rate', [0.001, 0.0005, 0.0001],
   group='optimizer')

DROPOUT_RATE = Linear('dropout_rate', 0.0, 0.5, 5,
   group='dense')

NUM_DIMS = Range('num_dims', 8, 32, 8, group='dense')

NUM_LAYERS = Range('num_layers', 1, 3, group='dense')

L2_NUM_FILTERS = Range('l2_num_filters', 8, 64, 8,
   group='cnn')

L1_NUM_FILTERS = Range('l1_num_filters', 8, 64, 8,
   group='cnn')
```

```
def model_fn():
    LR = Choice('learning rate', [0.001, 0.0005, 0.0001],
         group='optimizer')
    DROPOUT RATE = Linear('dropout_rate', 0.0, 0.5, 5,
         group='dense')
    NUM_DIMS = Range('num_dims', 8, 32, 8, group='dense')
    NUM_LAYERS = Range('num_layers', 1, 3, group='dense')
    L2_NUM FILTERS = Range('l2_num_filters', 8, 64, 8,
         group='cnn')
    L1_NUM FILTERS = Range('l1_num_filters', 8, 64, 8,
         group='cnn')
```

```
    model = Sequential()
    model.add(Conv2D(L1_NUM_FILTERS, kernel_size=(3, 3), activation='relu'))
    model.add(Conv2D(L2_NUM_FILTERS, kernel_size=(3, 3), activation='relu'))
    model.add(Flatten())
    for _ in range(NUM_LAYERS):
        model.add(Dense(NUM_DIMS, activation='relu'))
        model.add(Dropout(DROPOUT_RATE))
    model.add(Dense(10, activation='softmax'))
    model.compile(loss='categorical_crossentropy', optimizer=Adam(LR))

return model
```

```
tuner = Tuner(model_fn, 'val_accuracy' epoch_budget=500,
   max_epochs=5)

tuner.search(train_data,validation_data=validation_data)
```

https://colab.research.google.com/notebooks/welcome.ipynb#

Colaboratory에 오신 것을 환영합니다

파일 수정 보기 삽입 런타임 도구 도움말

공유

코드 텍스트 셀 셀 드라이브로 복사 연결 수정 중

목차 코드 스니펫 파일

- Colaboratory 소개
- 시작하기
- 추가 리소스
- 머신러닝 예제: Seedbank

섹션

Colaboratory에 오신 것을 환영합니다

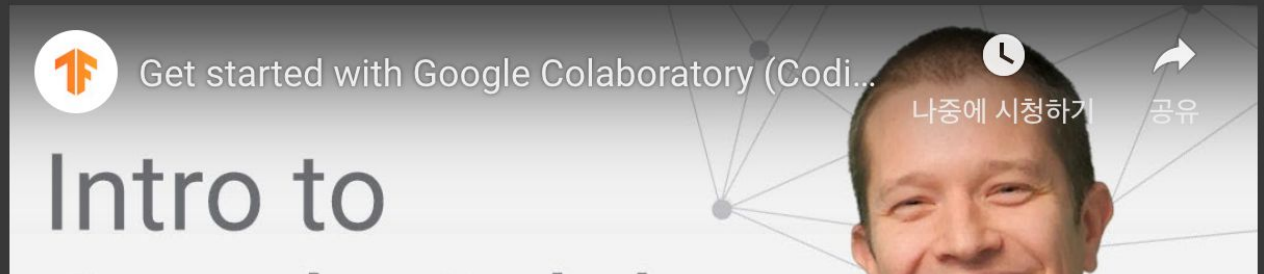
Colaboratory는 설치가 필요 없으며 완전히 클라우드에서 실행되는 무료 Jupyter 노트 환경입니다.

Colaboratory를 사용하면 브라우저를 통해 무료로 코드를 작성 및 실행하고, 분석을 저장 및 공유하며, 강력한 컴퓨팅 리소스를 이용할 수 있습니다.

[ ]

Colaboratory 소개

3분 길이의 동영상을 통해 Colaboratory의 주요 기능을 간단하게 알아보세요.



```
[ ]  1 !pip install git+https://github.com/keras-team/keras-tuner.git
```

```
Collecting git+https://github.com/keras-team/keras-tuner.git
  Cloning https://github.com/keras-team/keras-tuner.git to /tmp/pip-req-build-mg0
  Running command git clone -q https://github.com/keras-team/keras-tuner.git /tmp
Requirement already satisfied: tensorflow>=2.0.0-beta1 in /usr/local/lib/python3.
Requirement already satisfied: numpy in /usr/local/lib/python3.6/dist-packages (f
Requirement already satisfied: tabulate in /usr/local/lib/python3.6/dist-packages
Requirement already satisfied: terminaltables in /usr/local/lib/python3.6/dist-pa
Requirement already satisfied: colorama in /usr/local/lib/python3.6/dist-packages
Requirement already satisfied: tqdm in /usr/local/lib/python3.6/dist-packages (fr
Requirement already satisfied: requests in /usr/local/lib/python3.6/dist-packages
Successfully built Keras-Tuner
Installing collected packages: Keras-Tuner
  Found existing installation: Keras-Tuner 0.9.0.1561748737
    Uninstalling Keras-Tuner-0.9.0.1561748737:
      Successfully uninstalled Keras-Tuner-0.9.0.1561748737
Successfully installed Keras-Tuner-0.9.0.1561755703
```

```python
from tensorflow import keras
from tensorflow.keras import layers

import numpy as np

from kerastuner.tuners import RandomSearch
from kerastuner.engine.hypermodel import HyperModel
from kerastuner.engine.hyperparameters import HyperParameters
```

```
(x, y), (val_x, val_y) = keras.datasets.mnist.load_data()
x = x.astype('float32') / 255.
val_x = val_x.astype('float32') / 255.

x = x[:10000]
y = y[:10000]
```

```python
"""Basic case:
- We define a `build_model` function
- It returns a compiled model
"""

def build_model(hp):
    model = keras.Sequential()
    model.add(layers.Flatten(input_shape=(28, 28)))
    for i in range(hp.Range('num_layers', 2, 20)):
        model.add(layers.Dense(units=hp.Range('units_' + str(i), 32, 512, 32),
                               activation='relu'))
    model.add(layers.Dense(10, activation='softmax'))
    model.compile(
        optimizer=keras.optimizers.Adam(
            hp.Choice('learning_rate', [1e-2, 1e-3, 1e-4])),
        loss='sparse_categorical_crossentropy',
        metrics=['accuracy'])
    return model
```

```
"""Basic case:
- It uses hyperparameters defined on the fly
"""


tuner = RandomSearch(
    build_model,
    objective='val_accuracy',
    max_trials=5,
    executions_per_trial=3,
    directory='test_dir')
```

```
1 tuner.search_space_summary()
2
3 tuner.search(x=x,
4               y=y,
5               epochs=3,
6               validation_data=(val_x, val_y))
7
8 tuner.results_summary()
```

```
1 tuner.search_space_summary()
2
3 tuner.search(x=x,
4               y=y,
5               epochs=3,
6               validation_data=(va
7
8 tuner.results_summary()
```

Search space summary

|-Default search space size: 4

num_layers (Range)

|-default: 2
|-max_value: 20
|-min_value: 2
|-step: 1

units_0 (Range)

|-default: 32
|-max_value: 512
|-min_value: 32
|-step: 32

units_1 (Range)

|-default: 32
|-max_value: 512
|-min_value: 32
|-step: 32

learning_rate (Choice)

|-default: 0.01
|-values: [0.01, 0.001, 0.0001]

```
1 tuner.search_space_su
2
3 tuner.search(x=x,
4               y=y,
5               epochs=3
6               validati
7
8 tuner.results_summary
```

New model

Trial summary

Hp values:

|-learning_rate: 0.01
|-num_layers: 15
|-units_0: 288
|-units_1: 320

Execution 1/3

```
WARNING: Logging before flag parsing goes t
W0628 21:03:56.702659 140279821547392 depre
Instructions for updating:
Use tf.where in 2.0, which has the same bro
HBox(children=(IntProgress(value=0, max=313

HBox(children=(IntProgress(value=0, max=313
```

```
1 tuner.search_space_s
2
3 tuner.search(x=x,
4               y=y,
5               epochs=
6               validat
7
8 tuner.results_summar
```

Execution 3/3

```
HBox(children=(IntProgress(value=0, max=313),

HBox(children=(IntProgress(value=0, max=313),

HBox(children=(IntProgress(value=0, max=313),
```

Name	Best model	Current model
accuracy	0.5377	0.5377
loss	1.1953	1.1953
val_loss	1.2095	1.2095
val_accuracy	0.5524	0.5524

```
1 tuner.search_space_summar
2
3 tuner.search(x=x,
4               y=y,
5               epochs=3,
6               validation_d
7
8 tuner.results_summary()
```

Name	Best model	Current model
accuracy	0.7845	0.7845
loss	0.6646	0.6646
val_loss	0.6164	0.6164
val_accuracy	0.8097	0.8097

3/5 trials left

Name	Best model	Current model
accuracy	0.9287	0.9287
loss	0.2474	0.2474
val_loss	0.267	0.267
val_accuracy	0.9202	0.9202

2/5 trials left

```
1 tuner.search_
2
3 tuner.search(
4
5
6
7
8 tuner.results_
```

Name	Best model	Current model
accuracy	0.9287	0.1099
loss	0.2474	2.3019
val_loss	0.267	2.3018
val_accuracy	0.9202	0.1135

1/5 trials left

Name	Best model	Current model
accuracy	0.9348	0.9348
loss	0.2474	0.2494
val_loss	0.267	0.3198
val_accuracy	0.9229	0.9229

Hypertuning complete - results in test_dir/untitled_project

```
1 tuner.search_s
2
3 tuner.search(x
4               y
5               e
6               validation_data=(val_x, val_y))
7
8 tuner.results_summary()
```

```
val_accuracy 0.9229          0.9229
Hypertuning complete - results in test_dir/untitled_project
Results summary
|-Results in test_dir/untitled_project
|-Ran 5 trials
|-Ran 15 executions (3 per trial)
|-Best val_accuracy: 0.9229
```

You can use a HyperModel subclass instead of a model-building function

This makes it easy to share and reuse hypermodels.

A `HyperModel` subclass only needs to implement a `build(self, hp)` method.

```
from kerastuner import HyperModel


class MyHyperModel(HyperModel):

    def __init__(self, num_classes):
        self.num_classes = num_classes

    def build(self, hp):
        model = keras.Sequential()
        model.add(layers.Dense(units=hp.Range('units',
                                              min_value=32,
                                              max_value=512,
                                              step=32),
                               activation='relu'))
        model.add(layers.Dense(self.num_classes, activation='softmax'))
        model.compile(
            optimizer=keras.optimizers.Adam(
                hp.Choice('learning_rate',
```

```
        model.add(layers.Dense(self.num_classes, activation='softmax'))
        model.compile(
            optimizer=keras.optimizers.Adam(
                hp.Choice('learning_rate',
                          values=[1e-2, 1e-3, 1e-4])),
            loss='sparse_categorical_crossentropy',
            metrics=['accuracy'])
        return model


hypermodel = MyHyperModel(num_classes=10)

tuner = RandomSearch(
    hypermodel,
    objective='val_accuracy',
    max_trials=10,
    directory='my_dir',
    project_name='helloworld')

tuner.search(x, y,
             epochs=5,
             validation_data=(val_x, val_y))
```

Keras Tuner includes pre-made tunable applications: HyperResNet and HyperXception

These are ready-to-use hypermodels for computer vision.

They come pre-compiled with `loss="categorical_crossentropy"` and `metrics=["accuracy"]`.

```
from kerastuner.applications import HyperResnet
from kerastuner.tuners import Hyperband

hypermodel = HyperResnet(input_shape=(128, 128, 3), num_classes=10)

tuner = Hyperband(
    hypermodel,
    objective='val_accuracy',
    max_trials=40,
    directory='my_dir',
    project_name='helloworld')

tuner.search(x, y,
             epochs=20,
             validation_data=(val_x, val_y))
```

Computer Science > Machine Learning

Hyperband: A Novel Bandit-Based Approach to Hyperparameter Optimization

Lisha Li, Kevin Jamieson, Giulia DeSalvo, Afshin Rostamizadeh, Ameet Talwalkar

(Submitted on 21 Mar 2016 (v1), last revised 18 Jun 2018 (this version, v4))

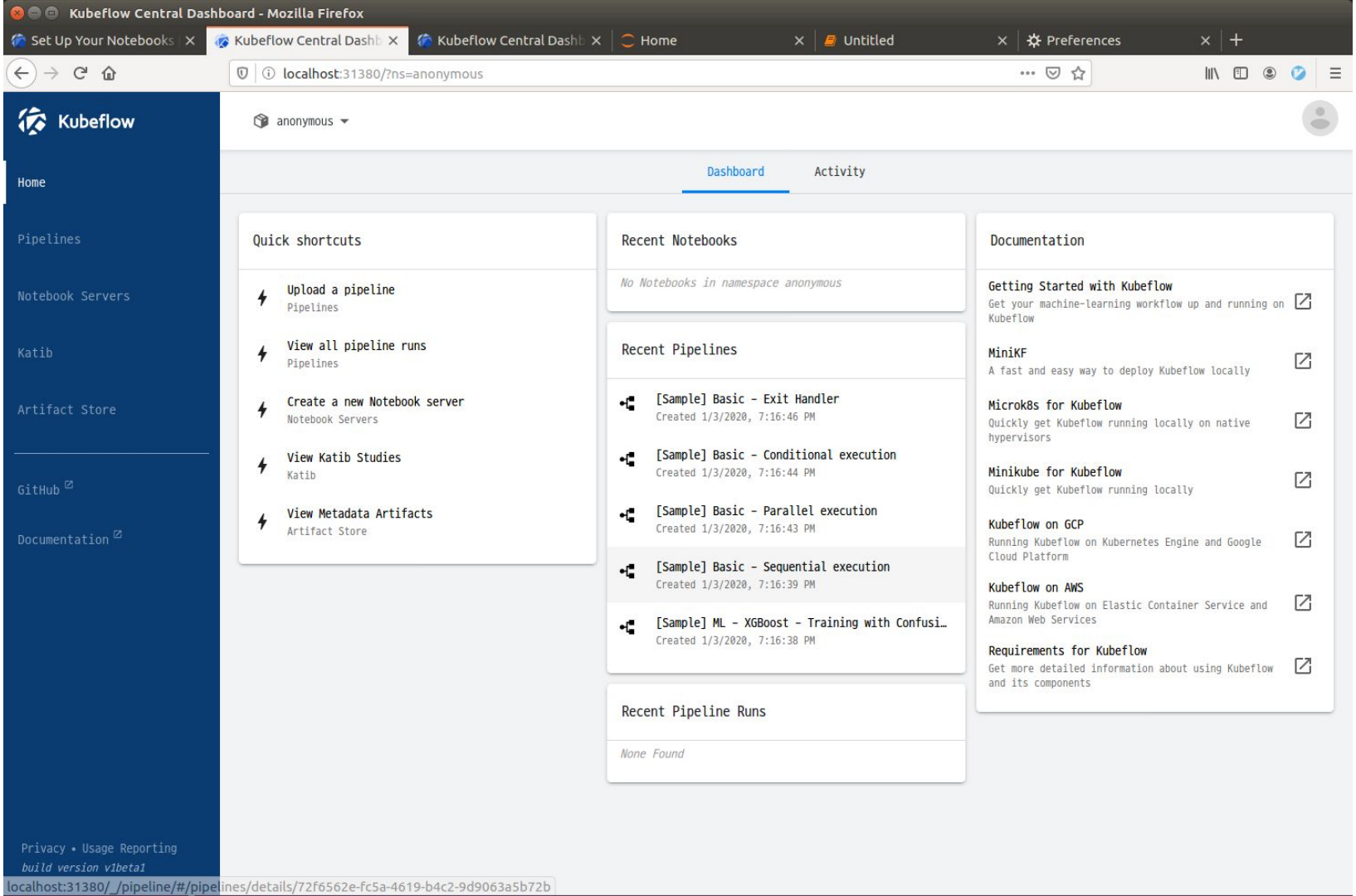
Performance of machine learning algorithms depends critically on identifying a good set of hyperparameters. While recent approaches use Bayesian optimization to adaptively select configurations, we focus on speeding up random search through adaptive resource allocation and early-stopping. We formulate hyperparameter optimization as a pure-exploration non-stochastic infinite-armed bandit problem where a predefined resource like iterations, data samples, or features is allocated to randomly sampled configurations. We introduce a novel algorithm, Hyperband, for this framework and analyze its theoretical properties, providing several desirable guarantees. Furthermore, we compare Hyperband with popular Bayesian optimization

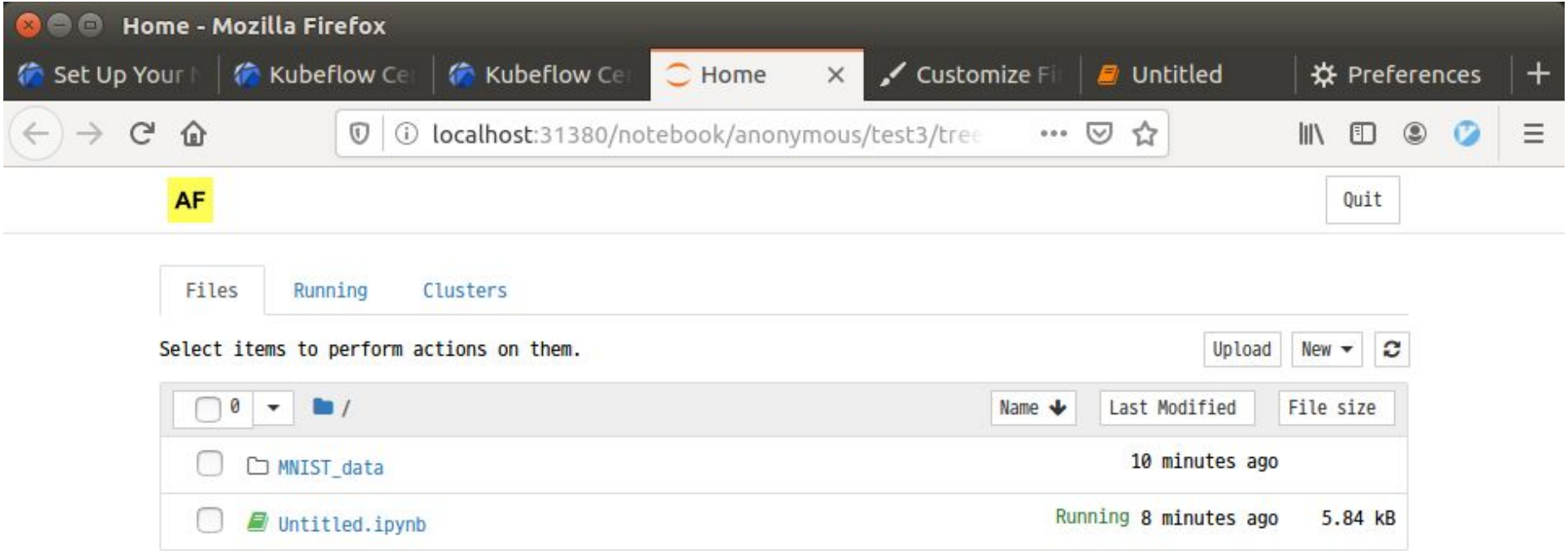
Keras Tuner includes pre-made tunable applications: HyperResNet and HyperXception

These are ready-to-use hypermodels for computer vision.

They come pre-compiled with `loss="categorical_crossentropy"` and `metrics=["accuracy"]`.

```
from kerastuner.applications import HyperResnet
from kerastuner.tuners import Hyperband

hypermodel = HyperResnet(input_shape=(128, 128, 3), num_classes=10)

tuner = Hyperband(
    hypermodel,
    objective='val_accuracy',
    max_trials=40,
    directory='my_dir',
    project_name='helloworld')

tuner.search(x, y,
             epochs=20,
             validation_data=(val_x, val_y))
```


Kubeflow Central Dashboard - Mozilla Firefox
Set Up Your Notebooks
Kubeflow Central Dashb
Kubeflow Central Dashb
Home
Untitled
Preferences
localhost:31380/?ns=anonymous
Kubeflow
anonymous
Home
Pipelines
Notebook Servers
Katib
Artifact Store
GitHub
Documentation
Privacy • Usage Reporting
build version v1beta1
Dashboard
Activity
Quick shortcuts
Upload a pipeline
Pipelines
View all pipeline runs
Pipelines
Create a new Notebook server
Notebook Servers
View Katib Studies
Katib
View Metadata Artifacts
Artifact Store
Recent Notebooks
No Notebooks in namespace anonymous
Recent Pipelines
[Sample] Basic - Exit Handler
Created 1/3/2020, 7:16:46 PM
[Sample] Basic - Conditional execution
Created 1/3/2020, 7:16:44 PM
[Sample] Basic - Parallel execution
Created 1/3/2020, 7:16:43 PM
[Sample] Basic - Sequential execution
Created 1/3/2020, 7:16:39 PM
[Sample] ML - XGBoost - Training with Confusi…
Created 1/3/2020, 7:16:38 PM
Recent Pipeline Runs
None Found
Documentation
Getting Started with Kubeflow
Get your machine-learning workflow up and running on Kubeflow
MiniKF
A fast and easy way to deploy Kubeflow locally
Microk8s for Kubeflow
Quickly get Kubeflow running locally on native hypervisors
Minikube for Kubeflow
Quickly get Kubeflow running locally
Kubeflow on GCP
Running Kubeflow on Kubernetes Engine and Google Cloud Platform
Kubeflow on AWS
Running Kubeflow on Elastic Container Service and Amazon Web Services
Requirements for Kubeflow
Get more detailed information about using Kubeflow and its components
localhost:31380/_/pipeline/#/pipelines/details/72f6562e-fc5a-4619-b4c2-9d9063a5b72b


Home - Mozilla Firefox
Set Up Your
Kubeflow Ce
Kubeflow Ce
Home
Customize Fi
Untitled
Preferences
localhost:31380/notebook/anonymous/test3/tre
AF
Quit
Files
Running
Clusters
Select items to perform actions on them.
Upload
New
0
/
Name
Last Modified
File size
MNIST_data
10 minutes ago
Untitled.ipynb
Running 8 minutes ago
5.84 kB

Untitled - Mozilla Firefox

Set Up Your N | Kubeflow Cer | Kubeflow Cer | Home | Untitled | Preferences

localhost:31380/notebook/anonymous/test3/not

jupyter Untitled (autosaved)

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

Run Code

In [1]:
```
from tensorflow.examples.tutorials.mnist import input_data
mnist = input_data.read_data_sets("MNIST_data/", one_hot=True)

import tensorflow as tf

x = tf.placeholder(tf.float32, [None, 784])

W = tf.Variable(tf.zeros([784, 10]))
b = tf.Variable(tf.zeros([10]))

y = tf.nn.softmax(tf.matmul(x, W) + b)

y_ = tf.placeholder(tf.float32, [None, 10])
cross_entropy = tf.reduce_mean(-tf.reduce_sum(y_ * tf.log(y), reduction_indices=[1]))

train_step = tf.train.GradientDescentOptimizer(0.05).minimize(cross_entropy)

sess = tf.InteractiveSession()
tf.global_variables_initializer().run()

for _ in range(1000):
  batch_xs, batch_ys = mnist.train.next_batch(100)
  sess.run(train_step, feed_dict={x: batch_xs, y_: batch_ys})

correct_prediction = tf.equal(tf.argmax(y,1), tf.argmax(y_,1))
accuracy = tf.reduce_mean(tf.cast(correct_prediction, tf.float32))
print("Accuracy: ", sess.run(accuracy, feed_dict={x: mnist.test.images, y_: mnist.test
```

```
WARNING: Logging before flag parsing goes to stderr.
W0115 09:14:22.851186 139876303353664 deprecation.py:323] From <ipython-input-1-f591bfcf84f5>:2: read_data_sets (from tensorflow.contrib.learn.python.learn.datasets.mnist) is deprecated and will be removed in a future version.
Instructions for updating:
Please use alternatives such as official/mnist/dataset.py from tensorflow/models.
W0115 09:14:22.852281 139876303353664 deprecation.py:323] From /usr/local/lib/python3.6/dist-packages/tensorflow/contrib/learn/python/learn/datasets/mnist.py:260: maybe_download (from tensorflow.contrib.learn.python.learn.datasets.base) is deprecated and will be removed in a future version.
Instructions for updating:
Please write your own downloading logic.
W0115 09:14:22.853549 139876303353664 deprecation.py:323] From /usr/local/lib/python3.6/dist-packages/tensorflow/contrib/learn/python/learn/datasets/base.py:252: _internal_retry.<locals>.wrap.<locals>.wrapped_fn (from tensorflow.contrib.learn.python.le
```

Untitled - Mozilla Firefox

Set Up Your N | Kubeflow Cer | Kubeflow Cer | Home | Untitled | Preferences

localhost:31380/notebook/anonymous/test3/not

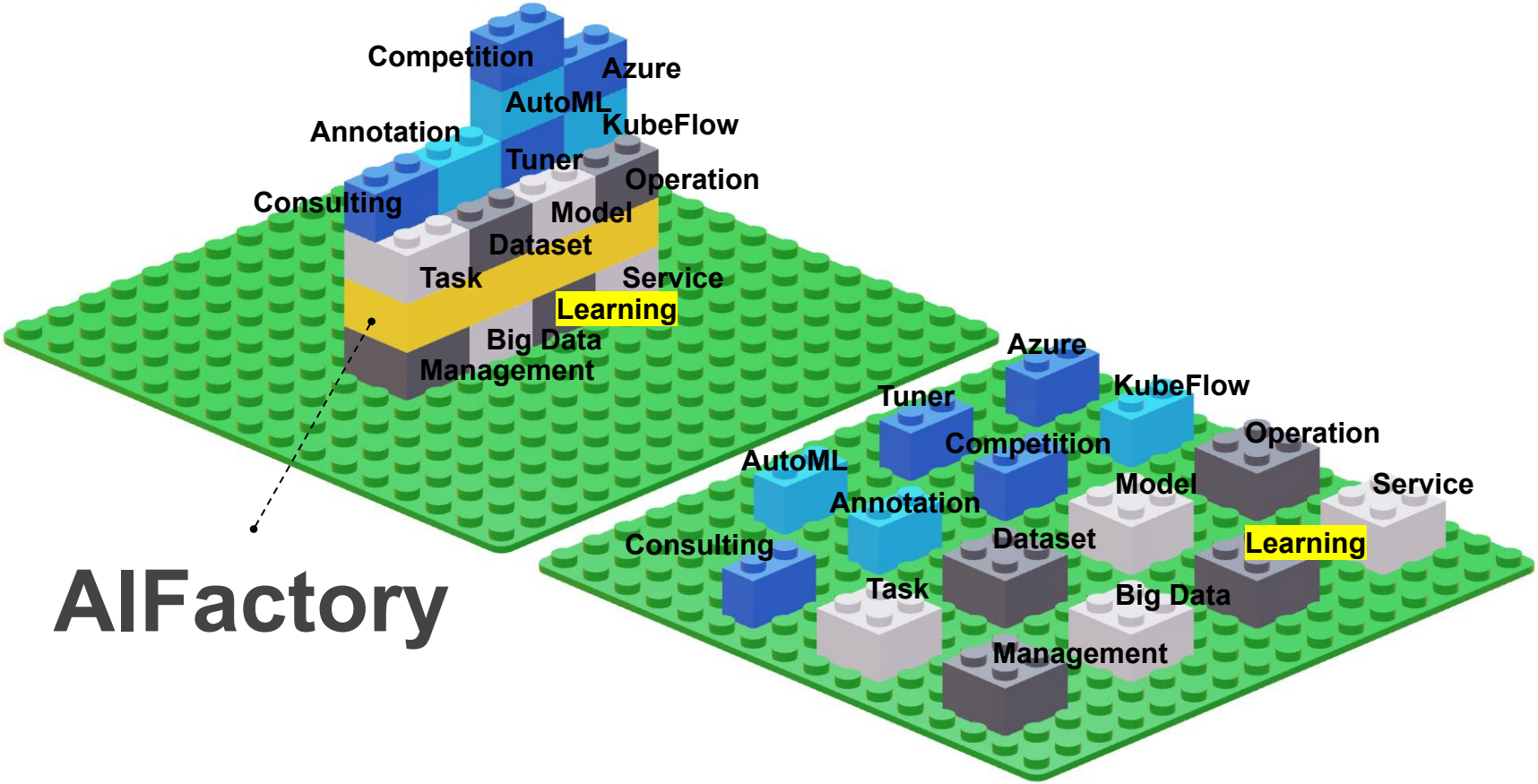
jupyter Untitled (autosaved)

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

Run Code

```
Instructions for updating:
Please use urllib or similar directly.
W0115 09:14:23.502009 139876303353664 deprecation.py:323] From /usr/local/lib/python3.6/dist-packages/tensorflow/contrib/learn/python/learn/datasets/mnist.py:262: extract_images (from tensorflow.contrib.learn.python.learn.datasets.mnist) is deprecated and will be removed in a future version.
Instructions for updating:
Please use tf.data to implement this functionality.
Successfully downloaded train-images-idx3-ubyte.gz 9912422 bytes.
Extracting MNIST_data/train-images-idx3-ubyte.gz
W0115 09:14:23.979117 139876303353664 deprecation.py:323] From /usr/local/lib/python3.6/dist-packages/tensorflow/contrib/learn/python/learn/datasets/mnist.py:267: extract_labels (from tensorflow.contrib.learn.python.learn.datasets.mnist) is deprecated and will be removed in a future version.
Instructions for updating:
Please use tf.data to implement this functionality.
W0115 09:14:23.988868 139876303353664 deprecation.py:323] From /usr/local/lib/python3.6/dist-packages/tensorflow/contrib/learn/python/learn/datasets/mnist.py:110: dense_to_one_hot (from tensorflow.contrib.learn.python.learn.datasets.mnist) is deprecated and will be removed in a future version.
Instructions for updating:
Please use tf.one_hot on tensors.
Successfully downloaded train-labels-idx1-ubyte.gz 28881 bytes.
Extracting MNIST_data/train-labels-idx1-ubyte.gz
Successfully downloaded t10k-images-idx3-ubyte.gz 1648877 bytes.
Extracting MNIST_data/t10k-images-idx3-ubyte.gz
W0115 09:14:24.612784 139876303353664 deprecation.py:323] From /usr/local/lib/python3.6/dist-packages/tensorflow/contrib/learn/python/learn/datasets/mnist.py:290: DataSet.__init__ (from tensorflow.contrib.learn.python.learn.datasets.mnist) is deprecated and will be removed in a future version.
Instructions for updating:
Please use alternatives such as official/mnist/dataset.py from tensorflow/models.
Successfully downloaded t10k-labels-idx1-ubyte.gz 4542 bytes.
Extracting MNIST_data/t10k-labels-idx1-ubyte.gz
Accuracy:  0.9024
```

In [ ]:


Competition
Azure
AutoML
KubeFlow
Annotation
Tuner
Operation
Consulting
Model
Dataset
Task
Service
Learning
Big Data
Management
Azure
KubeFlow
Tuner
Operation
Competition
AutoML
Model
Service
Annotation
Consulting
Dataset
Learning
Task
Big Data
Management

AIFactory

New
Home
Author
Notebooks
Automated ML
Designer
Assets
Datasets
Experiments
Pipelines
Models
Endpoints
Manage
Compute
Datastores
Data labeling

aifactory-smartfarm

Details Consume

Deployment state
Healthy

Compute type
AKS

Service ID
aifactory-smartfarm

Tags

Created on
12/24/2019 2:41:16 AM

Last updated on
12/24/2019 2:41:21 AM

Compute target
aifactory

REST endpoint
http://52.177.17.97:80/api/v1/service/aifactory-smartfarm/score

Key-based authentication enabled
false

Token-based authentication enabled
false

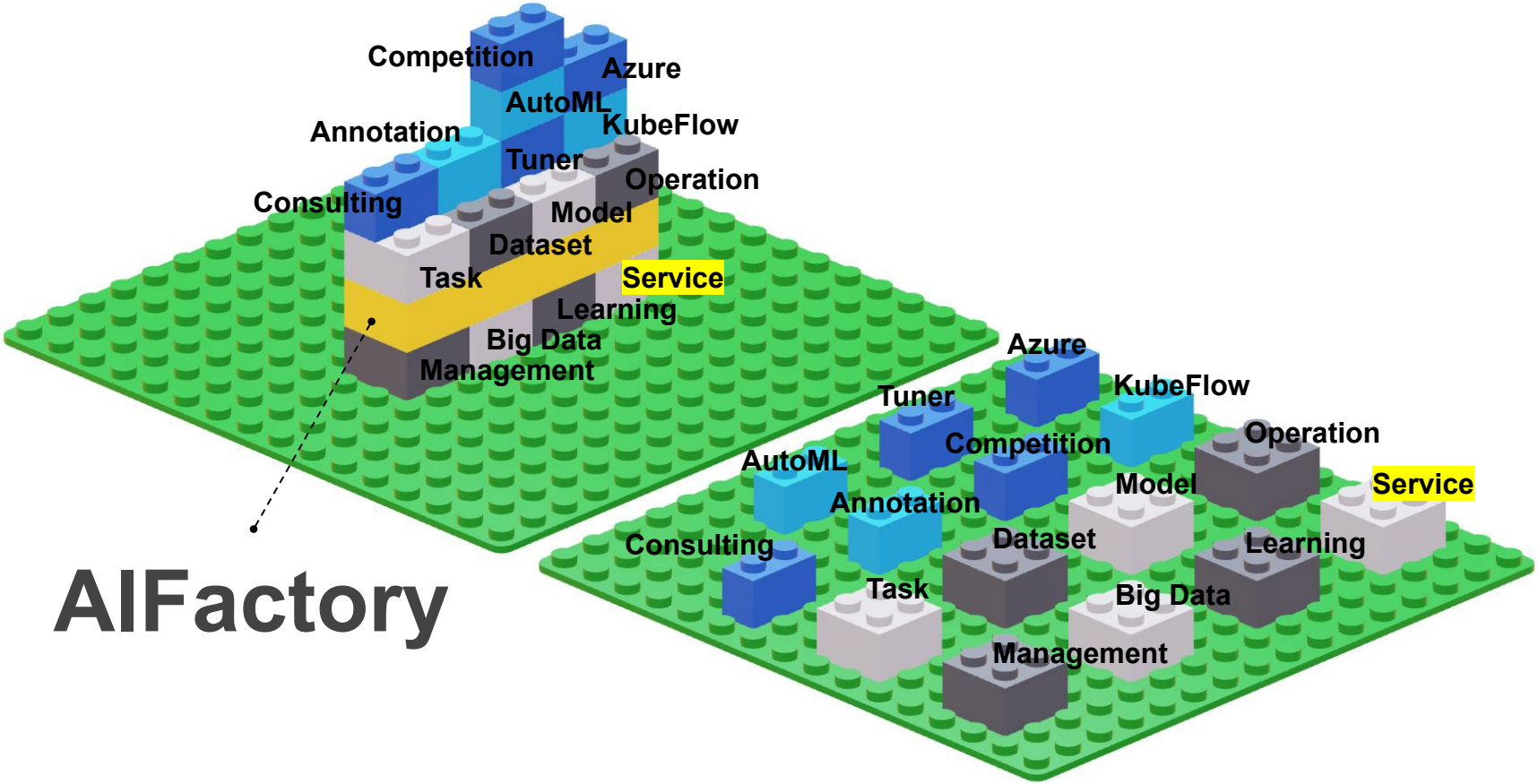
Swagger URI
http://52.177.17.97/api/v1/service/aifactory-smartfarm/swagger.json

CPU

In [2]:
```
import requests
import json

# URL for the web service.
scoring_uri = 'http://52.177.17.97:80/api/v1/service/aifactory-smartfarm/score'

#여기에 세가지 입력 변수의 값을 입력하면 됩니다.
data = {'data': [[15.0, 386.0, 12.0]]}

# Convert to JSON string.
input_data = json.dumps(data)

# Set the content type.
headers = {'Content-Type': 'application/json'}

# Make the request and display the response.
response = requests.post(scoring_uri, input_data, headers=headers)
print(response.status_code)
print(response.elapsed)
print(response.text)
```

```
200
0:00:00.093477
"{\"result\": [0.0]}"
```

In [ ]:


Competition
Azure
AutoML
KubeFlow
Annotation
Tuner
Operation
Consulting
Model
Dataset
Task
Service
Learning
Big Data
Management
Azure
KubeFlow
Tuner
Operation
Competition
AutoML
Model
Service
Annotation
Consulting
Dataset
Learning
Task
Big Data
Management

AIFactory

Artificial Intelligence Factory for You

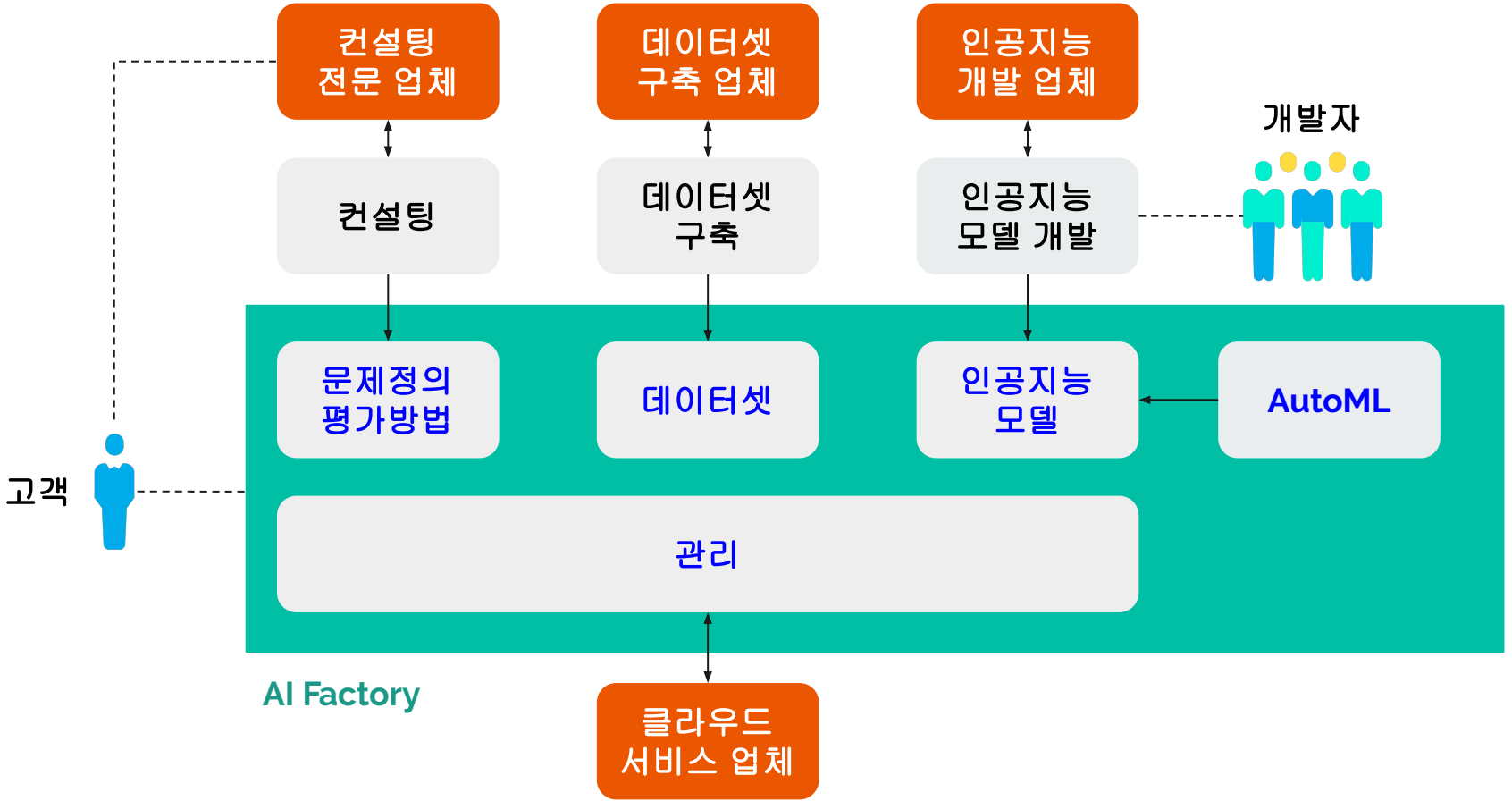
AIFactory

- 크라우드 소싱 협업
- **AutoML**
- 데이터셋 보안
- 모델 및 데이터셋 관리
- 클라우드 서비스

생태계 형성

유연한 협업과 올인원 서비스 제공

시장에서의 유연한 협업 방식



AI Factory 실제 사례

차세대 의료 진단기술을 위한 인공지능 기술 도입

New IVD system (VEUS)

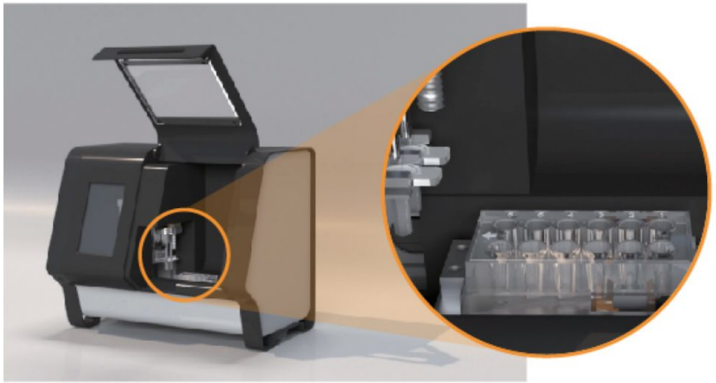
Versatile, Easy, and User-friendly IVD System

이지다이아텍의 VEUS 기술은 30분 이내에 빠르고 정확한 면역 분석을 수행할 수 있는 완전 자동 ALL IN ONE 시스템을 제공합니다.

효소 면역측정법(ELISA), 형광 면역측정법(FIA), 화학발광 면역측정법(Chemiluminescence Immunoassay, CLIA)과 같은 기존의 면역측정법은

정밀면역검사법으로 사용되고 있지만, 검사시간이 3~4시간 정도 소요되고, 검사장비가 고가이며, 검사를 수행할 전문 인력이 필요해서 중소병원에는 적합하지 않습니다.

VEUS 시스템은 30분내에 정확한 검사결과가 나오며, 누구나 손쉽게 조작할 수 있는 완전자동시스템이므로, 중소병원 및 응급실에 적합합니다.



1 짧은 예열시간 1~2분안에 37°C 도달

2 효과적인 Heating & Mixing 기술 자성입자의 특성을 살린 시스템

3 다중검지 시스템 독창적인 자성입자 코딩 시스템

4 All in one system 추가 검진기기가 필요없이 진단결과도출

Task

All | Hosted | Jointed

All
검색
검 색

2개의 Task

테스트 태스크입니다.

test task description
date(2019-11-07~)
model name : -, last score : 0, lab status: - / 2, next step target Score : -, next step Due Date : -, points : 0)

인공지능 기반 의료영상 분석

혈액검사의 결과 판독 시 자성입자끼리 뭉치거나 붙는 이미지 판독을 위한 해결책으로 AI를 이용하여 각 길이별로 구별을 진행하는 해결책을 찾고자 합니다.
date(2019-12-24~2019-12-28)
model name : -, last score : 0, lab status: - / 3, next step target Score : -, next step Due Date : -, points : 0)

인공지능 기반 의료영상 분석

혈액검사의 결과 판독 시 자성입자끼리 뭉치거나 붙는 이미지 판독을 위한 해결책으로 AI를 이용하여 각 길이별로 구별을 진행하는 해결책을 찾고자 합니다.

Data

데이터 셋 설명

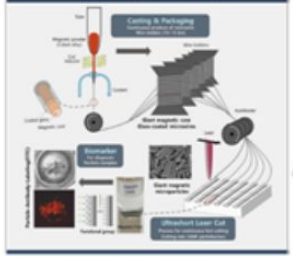
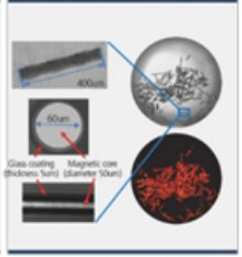
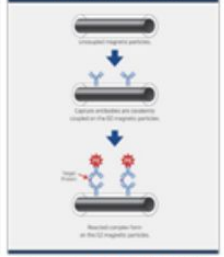
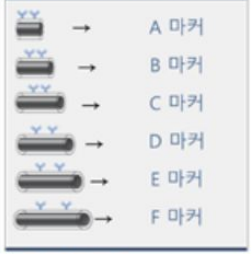
Edit

- 상기와 같이 혈액검사에 적합한 시스템 기술을 개발하여 현재 상용화를 진행하고 있으며 해당 기술은 빠른검사, 정확함, 고감도, 다중검지(multi-detection)의 기술적 특징을 가지며 그 중에서도 다중검지를 위하여 자성입자를 활용하고 있음.

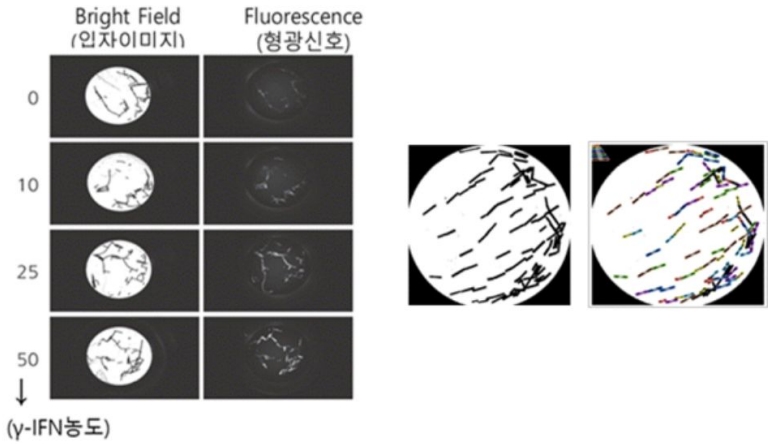
진단 영역	고객	주력 해외시장	아이템	주력제품
1. 고감도 /다중검지 제품	• 종합병원 • 응급센터 • 중소병원	• 선진국 (미국,유럽 등) / 중국	• 치매 진단 (2가지 이상 마커 검사)	
2. 고감도 제품	• 응급센터 • 중소병원	• 선진국 (미국,유럽 등) / 중국	• 고감도 심근경색진단 (hs-TnI)	
3. 동반 진단 제품	• 종합병원 • 중소병원	• 선진국 (미국,유럽 등) / 중국	• 유방암(신규마커) • 자궁경부암(신규마커)	

- 해당 시스템의 기술적 특징을 활용하여 상기와 같이 고감도의 다중검지 기능을 활용다면 보다 저렴한 비용과 짧은 시간을 가지고 신속하고 정확하게 진단을 내릴 수 있음.

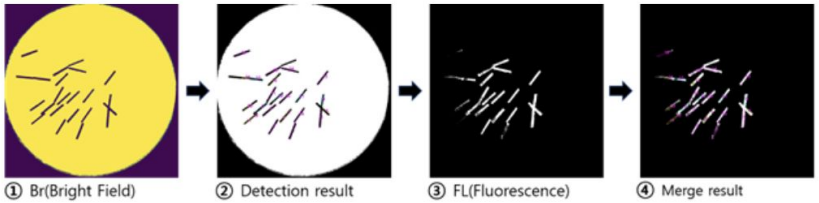
자성입자 제조 및 진단도입

자성입자 제조 방법	자성입자 현미경 사진	자성입자 면역반응	다중검지용 길이 코딩시스템
			
- Simple한 자성입자 제조공정 확립 - 다른 자성입자에 비해 재현성/생산성 개선 - Surface Chemistry 최적화 완성	- Rod Shaped Magnetic Particles - 거대 자성입자 (길이 200~500 um, 지름 60 um, Silica coating : 5 um) - 강한 Magnetic Power	- 업계최초로 반도체공정기술 도입 적용 - 신호 증폭효과 통한 고감도 기능	- 길이 코딩시스템 이용한 다중검지 기능 - Duplex, Triplex, Sixplex 등 다양한 Multiplex 가능

- 상기와 같이 자성입자를 제조하며 최대 7가지 길이로 구별하도록 하여 다중검지를 실시할 수 있으며 이를 위하여 형광촬영 된 이미지에서 각 길이별로 구별하여 형광값을 읽어와야 함.



- 혈액검사의 결과 판독은 상기 그림의 Bright Field의 이미지에서 각 길이별 자성입자를 구별 탐색한 정보를 가지고 형광신호 이미지에서 각 자성입자의 위치에 해당하는 형광신호 값을 판독하여 혈액검사 결과를 도출하도록 하고 있음.
- 이러한 방법을 통하여 자성입자 판독을 진행할 때의 어려움으로 자성입자의 자성 때문에 생기는 문제가 있음.
- 각 자성입자가 띄는 자성 때문에 자성입자끼리 뭉치거나 붙는 현상이 있으며 이러한 현상 때문에 현재 Bright Filed 에서의 각 길이별 자성입자의 구별에 어려움이 있어 이를 위한 해결책으로 AI를 이용하여 각 길이별로 구별을 진행하는 해결책을 찾고 있음.



- 현재 형광값의 산출은 상기 그림에서 ①에서와 같이 카트리지에 있는 자성입자의 BR 이미지를 획득하여 ②과 같이 자성입자의 위치를 찾고 ③과 같은 FL 이미지에 합쳐서 ④과 같은 최종 결과에서 원하는 자성입자의 형광 값을 읽어 들임.
- 상기의 사진은 500um 길이의 자성입자의 예이며 이 경우에도 ② 과 같이 검색하지 못하는 부분이 있으며 단일 크기가 아닌 7가지 길이의 자성입자를 혼합하는 경우 검색하지 못하는 부분이 상대적으로 많아지며 이의 대응을 위하여 AI를 이용하는 해법을 모색하고자 함.

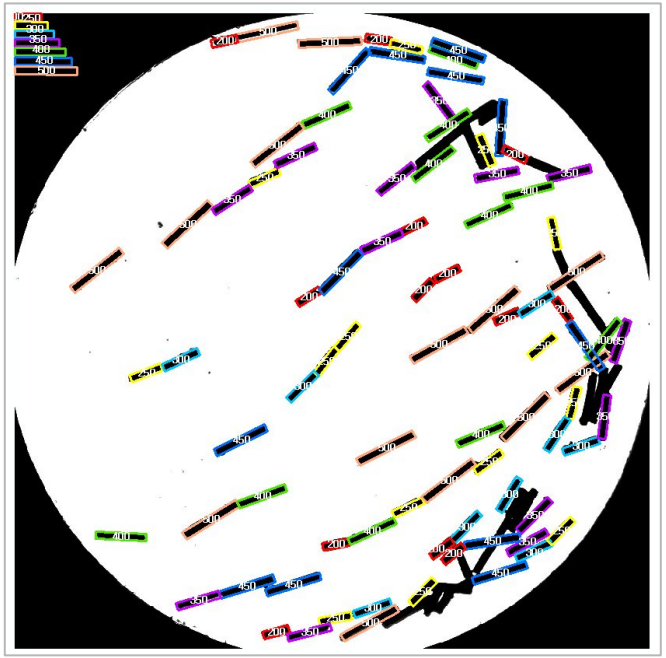
❖ 데이터 셋 분배

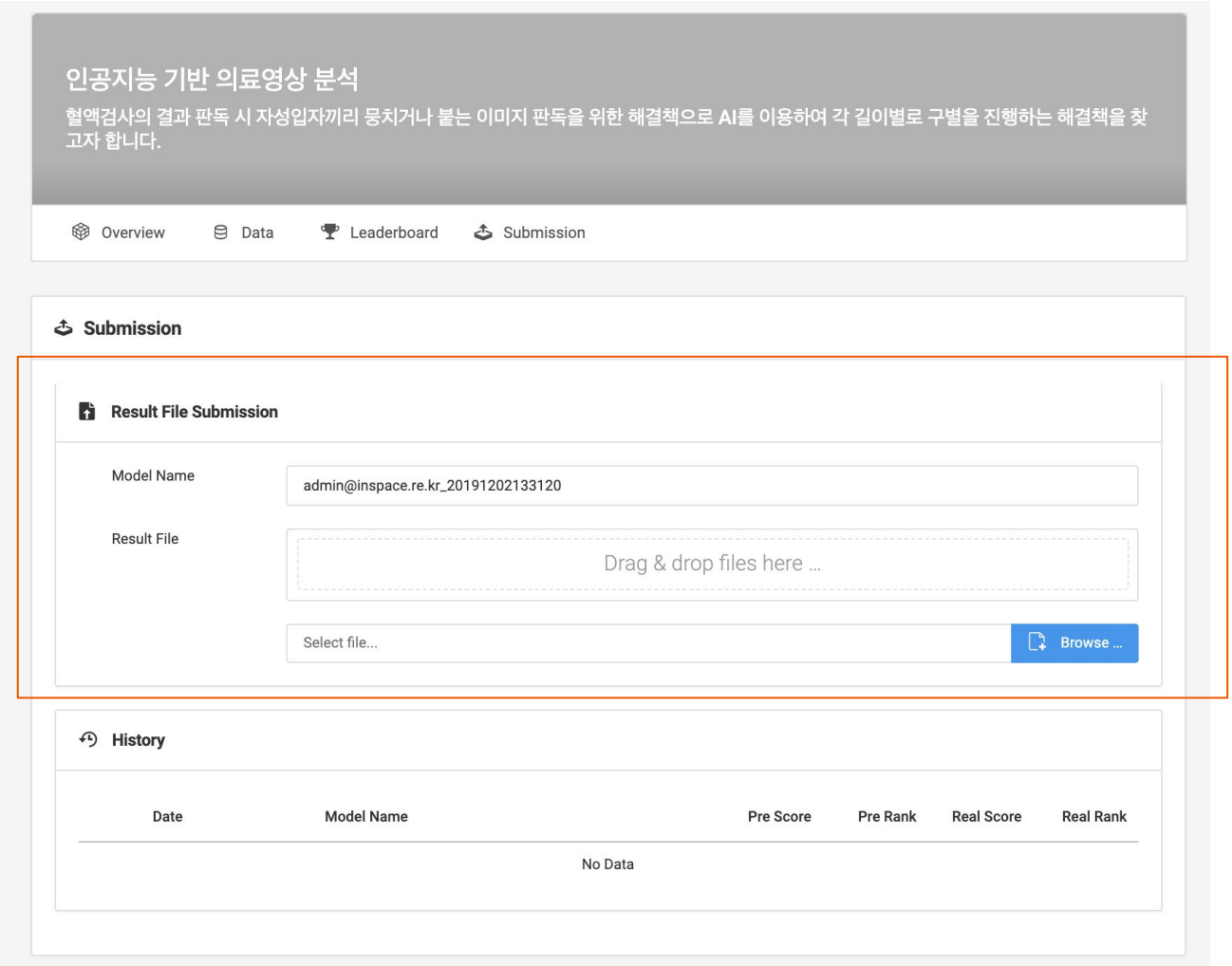
Baseline	Competition	Data Set
Train & Val	-	400
Test	Public	50
	Private	50

❖ 데이터 셋 구축

- ➢ Data set 500장 구축
- ➢ 라벨링 작업 (86/105)
- ➢ 작업시간:100분 ± α (86/105)
- ➢ 400장 Train & Val
- ➢ 100장 Test (Public:50장, Private:50장)

인원	1일 작업량(장)	총 작업량(장)	예상기간(일)
1	5	500	100
2	10	500	50
3	15	500	33
4	20	500	25

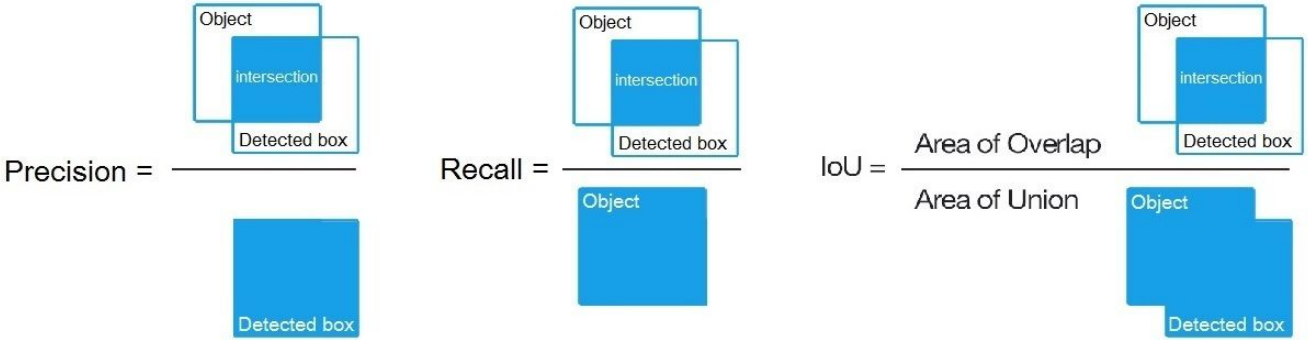




- 자신이 만든 모델의 결과 파일 제출 시 시스템에서 점수 계산
- 공정한 경쟁을 위해 정답지 유출 방지를 위한 문제 구성

평가방법

- 점수 계산 방식



- 점수 상황판 공개

Public Leaderboard Private Leaderboard

이 점수 현황은 테스트 데이터의 약 50% 로 계산됩니다.

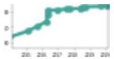
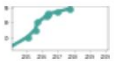
No.	UserName	TeamName	Score	Last	lap
1	전인택	-	82	2019-11-18 10:56:17	2/2
2	전인택	-	14	2019-11-07 16:55:59	1/2

AF

한국천문연구원
Korea Astronomy & Space Science Institute

출연연/산업체/대학 중심의 산업 AI 연구 교류 모임입니다.

Tasks

TREND	TASK	MODEL	DESCRIPTION	SCORE	STATUS
	최적설계	DesignNetv2	GAN based image generation model	0.76	Open
	Anomaly Detection	AnoNetv3	Auto Encoder	0.45	Close

MLOps KR

공개 그룹

정보
토론
공지
멤버
이벤트
동영상
사진
그룹 인사이트
함께 시청하기
그룹 관리
그룹 품질

이 그룹 검색

바로가기

fast.ai KR 2

변성윤
관리자 · 1월 20일

MLOps Korea는 주로 Serving, Model Management, Auto Feature Engineering, Hyperparameter tuning, AutoML에 대한 이야기를 다룹니다 🙂

관련 해외 글을 공유하거나, 번역하거나, 각종 프레임워크 튜토리얼, 영상 자료 등 어떤 것이든 환영해요 🙂

회원님, 이현호, 권순선, 외 29명

좋아요 댓글 공유하기

새 활동

변성윤님이 게시물을 공유했습니다.
관리자 · 12월 15일 오후 10:00

안녕하세요 🙂

머신러닝 엔지니어링 Best Practices에 대해 작성된 내용을 번역했습니다-!
내용을 보다보면 MLOps 관점으로 알아두면 내용도 있어 공유드립니다 🙂

변성윤님이 링크를 공유했습니다.
인기 게시물 작성 멤버 · 12월 15일 오후 9:56

안녕하세요 🙂

Google의 Research Scientist인 Martin Zinkevich가 작성하신 Rules of Machine Learning: Best Practices for ML Engineering 나름대로 번역하고 정리한 글 공유드립니다!

설명 설명을 추가해주세요.

그룹을 소개하세요.

그룹 유형

일반

위치

위치 추가

새 그룹 만들기

그룹을 이용하면 친구, 가족 및 팀원들과 훨씬 더 쉽게 공유할 수 있습니다.

그룹 만들기

최근 그룹 사진 모두 보기

추천 그룹 모두 보기

감사합니다.