AI

[2022년 #차]

문항 1) 다음은 데이터 imbalance 와 관련된 설명이다. 1) ~ 5) 각각에 대해, 괄호 안에 주어진 것 중 적절한 것을 선택하거나 괄호가 빈 경우 적절한 용어를 쓰시오. [6 점]

> 딥러닝 분류(classification) 문제 중 데이터 imbalance 는 정상 클래스(이하 majority 클래스)의 데이터 수와 이상 클래스(이하 minority 클래스)의 데이터 수 간에 큰 차이가 나는 경우를 나타낸다. 일반적으로 이러한 문제에서는 majority 클래스의 데이터 수가 minority 클래스의 데이터 수보다 현저하게 많으며, minority 클래스를 정확히 분류하는 것이 중요하다. 이러한 데이터 imbalance 로 인해 분류 경계선(decision boundary)의 위치가 1) (majority/minority) 쪽으로 치우쳐져 학습되기 쉬우며, 일반적으로 높은 2) (accuracy/recall/precision)을/를 보여 모델 성능에 대한 왜곡을 야기할 수 있다. 이를 해결하기 위한 방법 중 하나인 SMOTE(Synthetic Minority Oversampling Technique)는 minority 클래스 데이터를 3) (duplicate/random noise/interpolation)을/를 통해 증가시키는 방법으로 oversampling 으로 인한 4) (underfitting/overfitting/information loss/class overlap) 문제를 보완하는 데 효과적인 방법이다. 다만 oversampling 특성상 5) (　　)에 교란될 위험이 크다는 단점도 존재한다.

답안 작성: 1) _____. 2) _____. 3) _____. 4) _____. 5) _____.

(정답) 순서대로 minority, accuracy, interpolation, overfitting, 이상치(outlier)

(해설) data imbalance 상황에서 majority class 가 과잉대표되어 decision boundary 가 minority class 쪽으로 편향되며, 이러한 boundary 로 인해 majority class 를 예측하기 쉬워지므로 accuracy 가 높게 나오나 이는 일반적으로 minority class 를 구분하는 것이 중요한 이러한 문제에서 유효한 metric 이 되지 못한다. 이를 해결하기 위한 oversampling 기법인 SMOTE 는 minority class 의 데이터로부터 KNN(K-nearest neighbor)을 구하여 이들 샘플들을 interpolation 하여 새로운 샘플을 만들어낸다. 이런 방식은 random duplication 방식에서 주로 보이듯, 소수의 샘플이 과잉대표되어 발생하는 overfitting 을 완화시키는 효과가 있다. 다만 존재하는 sample 로부터 sample 을 생성해내는 oversampling 의 특성상 outlier 가 존재할 때 이로부터 합성된 데이터는 본래의 minority class 분포에서 크게 벗어나 생성되게 되며, 이는 decision boundary 를 교란시킬 수 있다.

[2022년 #차]

문항2) 철수는 다양한 가방 이미지(300x480 사이즈 컬러 이미지)를 가방의 색상에 따라 분류하는 딥러닝(Deep Learning) 모델을 개발하려고 한다. 해당 모델의 input size는 256x256x3이다. 가방 이미지에서 대상이 되는 가방은 중앙에 위치하고 있고 나머지 부분은 배경이다. [예시 이미지]와 같이 항상 정면의 이미지는 아니며 셀프 카메라 사진과 같이 반전되어 있는 사진들도 있다. 아래의 보기에서 사용 가능한 augmentation 방법을 모두 포함한 것을 고르시오. [4점]

[예시 이미지]

[보기]

- random crop(256,256)	- resize(256,256)	- center crop(256,256)
- rescale	- flip	- rotation
- grayscale		

① random crop, resize, flip, rotation
② resize, rescale, center crop, grayscale
③ center crop, rescale, flip, rotation
④ resize, rescale, center crop, rotation, grayscale
⑤ random crop, rescale, flip, rotation, grayscale

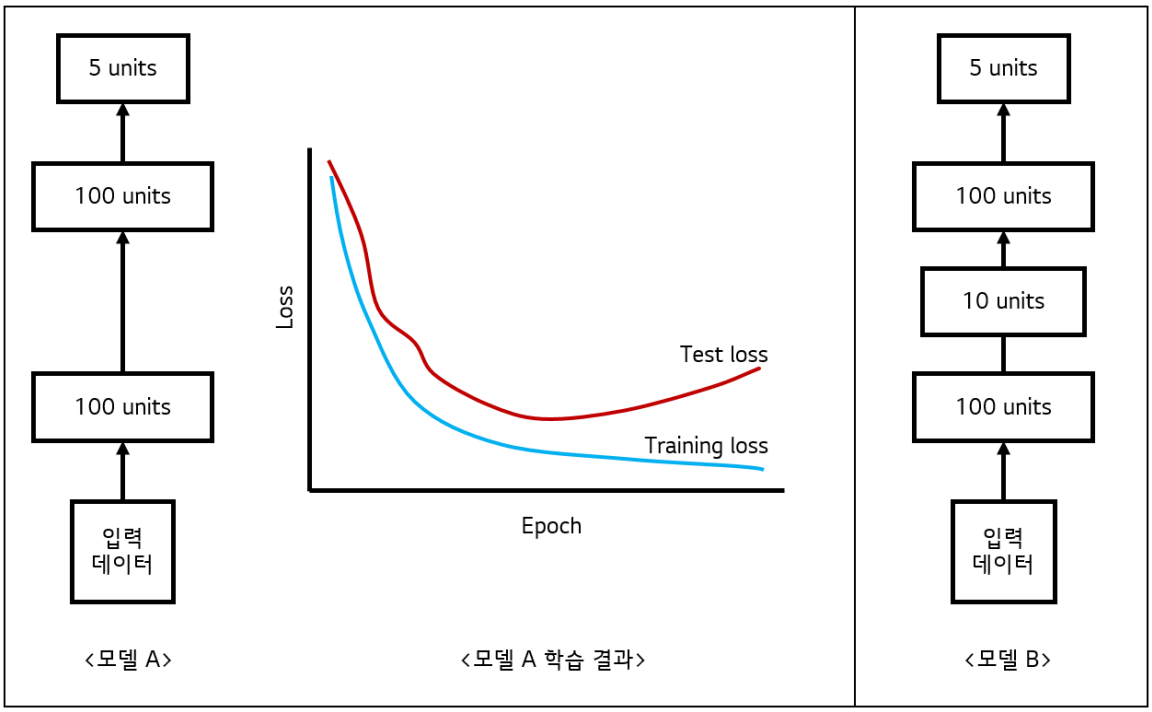
(정답) 3

(해설) 이미지 데이터에서 전처리를 진행할 때 효과적인 학습을 위해 rescale 이 필요하다. 문제 설명과 예시 이미지에 제시된 것을 고려할 때 target 이 있는 중앙 부분을 crop 하는 작업이 필요하므로 center crop 을 적용한다. 예시 이미지처럼 회전되어 있는 이미지 처리를 위해 rotation 과 셀프 카메라 사진처럼 반전되어 있는 이미지를 위해 flip 을 적용할 수 있다.
resize 의 경우 적용은 가능하나 정사각 이미지가 아닌 상황에서 정사각으로 resize 할 경우 형태의 변형이 있을 수 있어 문제의 [보기]에 제시된 것보다 더 큰 사이즈로 crop 한 후에 resize 하는 방법이 권장된다. 이 문제에서는 색상에 따라 분류하고 있으므로 grayscale 은 적용해서는 안 된다.

문항3) 철수는 입력 데이터를 다섯 개의 클래스로 분류(classification)하는 간단한 딥러닝(Deep Learning) 모델을 만들었다. 철수는 아래 <모델 A>와 같이 모델을 구성하였고, 학습 결과는 <모델 A 학습 결과>와 같았다.

철수는 <모델 A 학습 결과>를 보고, <모델 A>를 <모델 B>와 같이 수정했다. <모델 A>와 <모델 B>, 두 모델에 대한 설명 중 옳지 <u>않은</u> 것으로만 짝지어진 보기를 고르시오. [4점]

단, <모델 A>와 <모델 B>의 마지막 레이어(5 units)를 제외한 모든 레이어는 선형 활성화 함수(linear activation functions)를 사용한다고 가정하며, 그림에서 units는 각 레이어의 hidden nodes의 수를 나타낸다.



(가) <모델 B>는 <모델 A>보다 weights의 수가 더 적으므로 과대적합(overfitting)에 덜 취약하다.
(나) <모델 B>는 <모델 A>보다 입력 데이터에 내재하는 비선형적(non-linear) 관계를 표현하는데 더 유리하다.
(다) <모델 B>는 <모델 A>가 표현할 수 없는 함수*들도 표현할 수 있다.
(라) <모델 B>는 <모델 A>보다 더 compact representation을 배울 수 있다.

*함수: 딥러닝은 입력과 출력을 매칭하는 함수(관계)를 찾는 과정으로 간주할 수 있다.

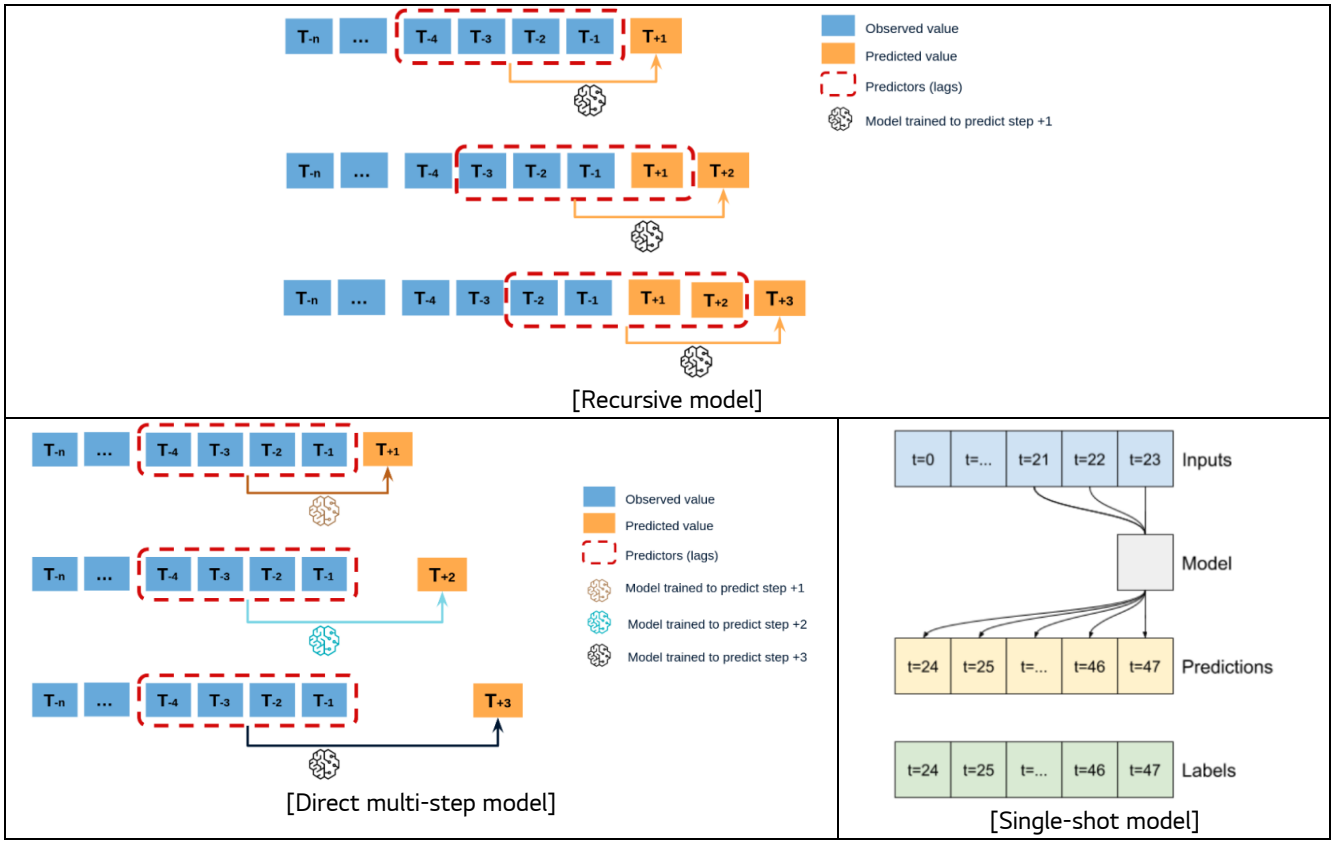
① (가), (나)
② (가), (다)
③ (가), (라)
④ (나), (다)
⑤ (나), (라)
⑥ (다), (라)

(정답) 4

(해설) (나) Hidden units 와 선형 활성화 함수로만 이루어져 있으므로 비선형적 관계를 표현하는데 더 유리하지 않다. (다) <모델 A>가 <모델 B>보다 표현력이 좋으며 더 많은 함수를 표현할 수 있다.

문항4) 철수는 제품 X의 과거 판매 이력을 기반으로 미래의 수요를 예측하는 딥러닝(Deep Learning) 모델을 개발하려고 한다. 판매 이력 데이터는 weekly 데이터이고 미래 1년을 예측하는 장기 모델을 개발하려고 한다. 철수는 예측하려는 구간의 전체 시계열이 한번에 예측되는 single-shot 모델과 모델이 단일 스텝 예측만 수행하고 출력이 입력으로 피드백되는 recursive 모델, 각 예측 대상 시점에 다른 모델을 사용하는 direct multi-step 모델을 고려하고 있다. 각 시계열 모델과 그에 따른 주의점에 대한 설명으로 옳지 <u>않은</u> 것을 고르시오. [4점]

[예시]



[Recursive model]

[Direct multi-step model]

[Single-shot model]

① Direct multi-step 모델은 예측해야 하는 step 수만큼 모델을 학습시켜야 한다.
② Single-shot 모델에는 CNN(Convolutional Neural Network) 계열, RNN(Recurrent Neural Network) 계열을 모두 사용 가능하다.
③ Recursive 모델의 경우 스텝이 길어질수록 점점 오차가 누적되므로 장기 예측 시 정확도가 떨어질 수 있다.
④ Single-shot 모델에서 CNN 계열을 사용할 경우 시간에 따른 변화를 학습할 수 있다.
⑤ Recursive 모델은 RNN 계열 모델만 사용 가능하다.

(정답) 5

(해설) Recursive 모델은 단일 time step 을 출력하도록 설계된 모든 모델에 적용할 수 있다. 단, output 값이 input 으로 피드백되어야 하므로 단일 input feature 를 사용하는 모델이 아닐 경우 output feature 를 제외한 나머지 input feature 의 미래구간 정보를 사용 가능하거나 모든 input feature 에 대한 단일 time step 을 예측하는 모델이어야 한다.

문항5) 철수는 Tensorflow 버전 2.2를 이용해 MNIST 데이터(숫자 0~9의 손글씨 이미지)를 0~9의 숫자 10가지로 분류(classification)하는 딥러닝(Deep Learning) 모델을 개발하려고 한다. 철수는 모델 학습을 위해 아래와 같은 [코드]를 작성하였다. 다음 중 아래 [코드]를 수행했을 때 발생하는 현상에 대한 설명으로 옳지 <u>않은</u> 것을 고르시오. [4점]

[코드]

```
import tensorflow as tf

num_labels = 10

def build_model(num_labels):
    inputs = tf.keras.Input(shape=(224, 224, 3), name='input', dytpe=tf.float32)

    test_model = tf.keras.applications.MobileNetV2(input_shape=(224, 224, 3), weights='imagenet',
                                                   include_top=False, pooling='avg')

    dense_layer = tf.keras.layers.Dense(num_labels, activation=None, use_bias=False,
                                        kernel_initializer='glorot_uniform')

    x = test_model(inputs)
    outputs = dense_layer(x)

    return tf.keras.Model(inputs=inputs, outputs=outputs)

def prepare_dataset():
    (x_train, y_train), (x_test, y_test) = tf.keras.datasets.mnist.load_data()

    # Preprocess the data (these are Numpy arrays)
    x_train = x_train.reshape(60000, 784).astype("float32") / 255.0
    x_test = x_test.reshape(10000, 784).astype("float32") / 255.0

    y_train = y_train.astype("float32")
    y_test = y_test.astype("float32")

    # Reserve 10,000 samples for validation
    x_val = x_train[-10000:]
    y_val = y_train[-10000:]
    x_train = x_train[:-10000]
    y_train = y_train[:-10000]

    return x_train, y_train, x_val, y_val
x_train, y_train, x_val, y_val = prepare_dataset()
test_model = build_model(num_labels)

optimizer = tf.keras.optimizers.Adam(learning_rate=1e-3)
loss = tf.keras.losses.CategoricalCrossentropy(from_logits=False)
metrics = tf.keras.metrics.CategoricalAccuracy()
test_model.compile(
    optimizer=optimizer,
    loss=loss,
    metrics=[metrics],
)
```

참고 사항: 아래는 MNIST 데이터의 일부를 나타낸다.

```
x_train[:10], y_train[:10]
```

```
(array([[0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.],
        ...,
        [0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.]], dtype=float32),
 array([5., 0., 4., 1., 9., 2., 1., 3., 1., 4.], dtype=float32))
```

① [코드]의 test_model.compile 코드는 오류 없이 수행된다.
② prepare_dataset에 y에 대한 one-hot encoding을 추가하면 accuracy를 정상적으로 계산할 수 있다.
③ prepare_dataset에 y에 대한 one-hot encoding을 추가하면 loss를 정상적으로 계산할 수 있다.
④ metrics를 SparseCategoricalAccuracy로 변경해야 한다.
⑤ test_model에 대한 학습 수행 시 오류가 발생한다.

(정답) 3

(해설) ③build_model 에서 dense_layer 에 activation=None 으로 되어 있으므로 one-hot encoding 적용 외에 loss function 에 추가로 from_logits=True 옵션을 적용해야 한다. Accuracy 는 argmax 를 사용하기 때문에 logits 여부에 영향을 받지 않는다.

문항6) 철수는 영화 리뷰를 수집하여 주제를 분류하는 작업을 수행하려 한다. 이를 위해 총 n개의 영화 리뷰를 수집하였고, 각각의 리뷰에 대해 최대 3개의 라벨을 붙인 데이터셋을 확보하였다. 이때 전체 라벨은 감동, 슬픔, 화남 등 총 m개의 주제(m=30)를 포함하고 있다. 철수가 아래 주어진 단계를 따라 딥러닝(Deep Learning) 모델을 설계하고 적용하였다고 할 때, 옳지 <u>않은</u> 방법으로 짝지어진 것을 고르시오. [4점]

(가) 분류 문제이므로 categorical cross-entropy loss를 적용하였다.
(나) 모델의 최종 layer의 활성화 함수(activation function)로 sigmoid function을 사용하고 최종 추론 시에 리뷰마다 임계값(threshold)을 넘는 주제 중 수치가 높은 상위 3개까지는 True 값을 부여하고, 나머지 값은 False 값을 부여한다.
(다) 전체 데이터셋에 대해, 각 리뷰가 어떤 주제를 가지고 있는지에 대한 전체 추론 결과(n by m matrix)와 전체 ground truth label 데이터(n by m matrix)를 element-wise로 총 n*m개의 값을 비교하여 90% 정확도(accuracy)를 달성하였다.
(라) 리뷰의 길이가 비정형적이므로 RNN과 Attention을 사용하였다.

① (가), (나)
② (가), (다)
③ (나), (다)
④ (나), (라)
⑤ (가), (다), (라)
⑥ (가), (나), (라)

(정답) 2

(해설) (가) 멀티 레이블 문제이므로 binary cross-entropy 를 사용해야 한다. (다) 30 개의 주제 중 최대 3 개가 달릴 수 있기 때문에 모든 label 이 달리지 않게 추론하여도 리뷰당 3/30 의 비율로만 틀리게 된다. 따라서 위와 같이 element-wise 방식으로 계산한 accuracy 는 주어진 문제 상황에 적합하지 않다. 위 경우에는 row-wise 로 계산하거나 f-beta score 에서 beta 값을 조절하여 적용하는 것이 옳다.

문항7) 철수는 강아지와 고양이의 이진 분류(binary classification) 문제를 풀기 위해 Tensorflow 버전 2.2로 작성된 아래 [코드]의 딥러닝(Deep Learning) 모델을 만들었다. 철수는 모델의 최종 출력값(return으로 나오는 값)이 0.5 이상인 이미지는 모두 '강아지'로 분류하기로 하고 학습을 진행하였다. 이때, 아래 [코드]를 적용하여 만든 모델에서 모든 입력데이터를 '강아지'로 분류하는 문제가 발생하였다. 주어진 [코드]에서 틀린 부분과 적절한 수정 방안으로 옳은 것을 고르시오. [4점]

[코드]

```
class Cat_Dog_Model(Model):
    def __init__(self):
        super(Cat_Dog_Model, self).__init__()
    ① self.flatten = Flatten()
    ② self.pooling = MaxPooling2D((2,2))
        self.conv_1 = Conv2D(8, 7, padding='same', activation='relu')
    ③ self.conv_2 = Conv2D(16, 5, padding='same', activation='relu')
        self.hidden_1 = Dense(15, activation='relu')
    ④ self.hidden_2 = Dense(1, activation='relu')
    ⑤ self.output = Activation('sigmoid')

    def call(self, x):
        x = self.conv_1(x)
        x = self.pooling(x)
        x = self.conv_2(x)
        x = self.pooling(x)
        x = self.flatten(x)
        x = self.hidden_1(x)
        x = self.hidden_2(x)
        return self.output(x)

model = Cat_Dog_Model()
```

① Flatten 대신 Dense layer로 변경한다.
② pooling size를 3,3으로 변경한다.
③ Conv2D의 kernel size를 3으로 변경한다.
④ activation=None으로 변경한다.
⑤ sigmoid 대신 softmax를 사용한다.

(정답) 4

(해설) Relu activation 함수의 결과는 항상 0 보다 같거나 크므로 이후 self.output 에서 sigmoid 를 태우면 항상 0.5 이상의 값을 output 으로 출력하게 되며 이는 모든 입력을 개로 분류하게 한다. 이 문제에서는 relu 와 sigmoid 가 연속으로 적용되어 문제가 생기므로 Dense layer 의 activation 을 None 으로 변경하면 된다.

문항 8) 다음은 딥러닝(Deep learning)에서 활용되는 여러 방법들과 관련된 서술이다. 적절한 것을 모두 고르시오. (2 개) [5 점]

① 이미지 데이터 추론 시, 일반적으로 한 장의 이미지로는 정규화를 할 수 없으므로 배치 정규화(batch normalization) layer는 스킵한다.
② 모델의 validation set 정확도가 train set 정확도와 비교하여 현저히 낮을 때, batch normalization은 모델의 일반화(generalization)에 긍정적인 효과를 준다.
③ Softmax 함수는 예측값을 확률 분포(probability distribution)의 형태로 정규화(normalize)하므로, n차원 벡터 y에 대해 y의 softmax 결과값과 cy (c는 임의의 실수)의 softmax 결과값은 동일하다.
④ 표준화(standardization)를 하기 위해 전체 데이터셋에 대해 평균(mean)과 표준편차(standard deviation)를 계산한 후 train/validation/test split을 수행한다.
⑤ 모델 학습 시 Early stopping은 모델의 variance를 증가시키는 경향이 있다.
⑥ Batch size 를 128 에서 32 로 줄이면 더 좋은 일반화(generalization)를 위한 regularizing 효과를 보인다.

(정답) 2,6

(해설) ①추론 시에도 batch normalization layer 는 스킵하지 않으며 저장해 놓은 평균/분산값을 활용한다. ③Softmax 함수에는 exponential 함수가 포함되어 있어 실수배를 하더라도 결과값은 동일하지 않을 수 있다. ④split 을 하기 전에 정규화를 수행하면 test 의 정보가 학습 단계에 유입될 수 있다. ⑤Early stopping 은 regularization 의 일종으로 variance 를 낮추는 경향이 있다.

문항 9) 다음 수식들은 딥러닝(Deep Learning) 분야에서 사용하는 loss 함수의 형태 및 regularization 기법 중 L1 regularization 과 L2 regularization 을 나타낸다. 두 수식을 참고하여 [보기]의 1) ~ 3)에 대해, 주어진 설명이 적절하면 O 로, 적절하지 않으면 X 로 답안을 작성하시오. [5 점]

$$L = L_0 + \lambda\Omega(w), \qquad \lambda \geq 0$$

<식1> L1 또는 L2 regularization이 적용된 loss 함수의 형태
(L_0는 regularization 적용 전 loss 값)

$$\text{L1: } \Omega(w) = \frac{1}{n}\sum_{w}|w| \qquad \text{L2: } \Omega(w) = \frac{1}{2n}\sum_{w}w^2$$

<식2> L1 regularization 수식 **<식3>** L2 regularization 수식

[보기]

1) L1 regularization 을 적용하면 일부 weight 를 0 으로 만들 수 있다.
2) L1 과 L2 regularization 은 convex 함수인 loss 가 최소값에 더욱 쉽게 가까워지도록 돕는 역할을 한다.
3) L1 regularization 을 적용하면 weight 값이 큰 경우가 weight 값이 작은 경우보다 weight 의 변화폭이 더 크다.

답안: 1) _____. 2) _____. 3) _____.

(정답) O, X, X

(해설) 1) L1 regularization 은 L2 regularization 과 달리 일부 weight 를 0 으로 만들 수 있다. 2) L1 과 L2 regularization 은 loss 가 최소값에 가까워지는 것을 어렵게 하여 regularizer 의 역할을 수행한다. 3) L1 의 경우 동일한 폭으로 감소한다.

문항10) 철수는 공장 환경에서 로봇 에이전트(agent)가 자율주행을 하며 작업을 수행하는 시스템을 강화학습(Reinforcement Learning)을 통해 구현하고자 한다. 공장 안에는 재료를 처리하는 기기들이 곳곳에 배치되어 있고, 로봇은 재료를 가지고 이리저리 옮겨다니며 각 기기에서 정해진 공정을 수행하여 최종 산출물을 출하 장소로 옮기게 된다. 로봇은 처음 작업에 착수할 때 공정을 위해 방문해야 하는 기기 목록을 받게 되고, 곳곳에 있는 동일한 기기 중 가깝거나 순서가 밀리지 않는 기기를 찾아 공정을 수행하도록 학습이 되어야 한다.

철수는 위 시나리오를 고려하여 강화학습을 위한 보상(reward) 체계를 설계하였다. 철수는 로봇의 시간 지연을 막기 위해 이동 횟수에 따라 페널티(penalty)를 부여하고 최종 산출물이 발생했을 때 보상을 부여하였다. 또한 이를 수행하기 위해 로봇 하나를 학습 대상으로 삼고 나머지 로봇들은 기존에 공장에서 사용하던 룰 기반 알고리즘(rule-based algorithm)으로 동작하게 하여 학습을 진행하였다. 하지만 이 모델에서 학습이 실패하여 로봇이 방황하거나 정지해 있는 상황이 관찰되었다. 이를 보완하기 위해 시도해 볼 만한 대안으로 옳은 것만으로 짝지어진 것을 고르시오. [4점]

(가) 중간 공정을 수행할 때마다 소량의 보상을 주도록 보상을 재설계한다.
(나) 사람이 로봇을 조종하여 필요한 공정을 수행하게 하고 이를 메모리에 저장하여 학습에 활용하게 한다.
(다) 모든 로봇에 동일한 모델을 도입하여 비동기적으로(asynchronous) 동시에 학습이 진행되도록 한다.
(라) 이동 횟수 대신 시간 지연 자체에 페널티를 부과하도록 보상을 재설계한다.
(마) 학습 초기에 랜덤(random) 행동의 비율을 늘려 좀 더 공장을 넓게 돌아다니도록 유도한다.

① (가), (나)
② (가), (다), (라)
③ (나), (다), (라), (마)
④ (가), (나), (라), (마)
⑤ (가), (나), (다), (라), (마)

(정답) 4

(해설) (가) 최종 산출에만 보상을 주어 보상이 희소(Sparse)하게 되어 학습이 진행되지 않은 문제이므로 중간 보상을 주어 보상의 희소성 문제를 완화시킬 수 있다. (나) 희소 보상의 문제는 모사 학습을 통해서도 해결 가능하다. (다) 비동기적 동시 학습은 보상의 희소 문제와 상관없다. 이는 다중 Agent 환경에서 상호 간섭에 대한 최적 전략의 학습에 필요하다. (라) 로봇이 정지해 있는 경우가 관찰되었던 것은 작업 수행을 통한 보상 획득보다 이동 비용으로 인한 손실이 커졌기 때문이다. 정지해 있더라도 비용이 발생하면 보상을 받고 에피소드를 끝마치기 위해 탐색이 촉진된다. (마) 희소 보상의 문제이므로 탐색 범위를 넓혀 최종 보상을 획득한 경험이 늘어나면 학습에 도움이 될 수 있다.

문항11) 다음은 깊은 신경망(Fully Connected Deep Neural Networks)에 적용할 수 있는 2가지 정규화(Regularization) 전략이다. [보기]의 (가)~(바)에 대해 Bias와 Variance가 각각 어떻게 되는지, 다음 3가지 범례로 작성하시오. 단, 답안 작성은 (가)~(바)의 순서를, 쉼표로 구분하여 작성한다. [5점]

[범례] 변화없음, 증가함, 감소함

[답안 작성 예((가)~(바)의 순서): 증가함, 증가함, 증가함, 증가함, 증가함, 증가함]

[보기]

전략	Bias	Variance
Dropout 사용	(가)	(나)
더 많은 학습 데이터 추가	(다)	(라)
Weight decay 사용	(마)	(바)

답안: _____.

(정답) 증가함, 감소함, 변화없음, 감소함, 증가함, 감소함

(해설) Dropout 사용 시 Bias 는 증가하고 Variance 는 감소한다. 더 많은 학습 데이터를 추가하는 경우에는 유사하게 Variance 는 감소하지만 Bias 는 변화없다. Weight decay 는 dropout 과 bias, variance 측면에서 유사한 효과를 가진다.

문항12) 철수는 Tensorflow 버전 2.2를 이용해 기판 이미지의 불량 여부를 판단하는 딥러닝(Deep Learning) 모델을 개발했다. 이 모델을 실제 현장에 적용할 때 추론 속도가 현장에서 요구하는 기준치보다 느려 @tf.function 기능을 사용해 개선하려고 한다. 입력 이미지는 현장의 장비에 따라 width와 height가 다를 수 있으며 여러 장비에서 얻어진 이미지를 모아 추론을 진행한다. 철수는 모델의 input size를 224x224x3으로 하고 @tf.function을 적용하여 아래 [코드]를 작성했다. 다음 중 [코드]에 대한 설명으로 옳지 <u>않은</u> 것을 고르시오. (단, model_path와 image_path는 별도로 지정된 값이다.) [5점]

[코드]

```python
import tensorflow as tf
import os

loaded_model = tf.keras.models.load_model(model_path, custom_objects={'tf': tf})

@tf.function
def evaluate_model(image):
    image = tf.image.resize(image, [224,224])
    image = tf.clip_by_value(image, 0, 255)
    image /= 255.0
    image = tf.expand_dims(image, axis=0)
    output = loaded_model(image, training=False)

    return output

image_list = os.listdir(image_path)
output_list = []
for i in range(len(image_list)):
    image_file_path = os.path.join(image_path,image_list[i])
    if os.path.exists(image_file_path):
        with open(image_file_path, 'rb') as f:
            image = f.read()
    image = tf.cast(tf.io.decode_image(image, channels=3, expand_animations=False), dtype=tf.float32)

    out = evaluate_model(image)
    output_list.append(out)
```

① tf.function을 적용할 메소드 내부에서 tf.expand_dims 대신 np.expand_dims를 사용하면 오류가 발생한다.
② 모델과 관련된 옵션을 조절하는 용도로 evaluate_model에 argument를 추가할 경우 해당 값을 Tensor로 입력해야 argument 값 변경 시 트레이싱(tracing)에 영향을 주지 않는다.
③ evaluate_model.get_concrete_function을 사용하면 고정으로 트레이싱된 evaluate_model 함수를 얻을 수 있어 evaluate_model을 이것으로 대체하는 것이 좋다.
④ evaluate_model의 input으로 들어가는 image의 사이즈가 달라지면 tf.function의 재트레이싱이 발생하므로 resize 작업을 evaluate_model 수행 이전에 하는 것이 바람직하다.
⑤ tf.function을 적용한 메소드는 Tensor를 input으로 사용 가능하므로 tf.io.decode_image 사용 시 별도의 type 변환이 필요하지 않다.

(정답) 3

(해설) ③get_concrete_function 을 사용하면 재트레이싱은 방지할 수 있으나 이 문제의 경우 image 의 사이즈가 다양하기 때문에 get_concrete_function 사용 시 지정한 사이즈의 이미지와 다른 사이즈의 이미지가 들어올 경우 오류가 발생하게 된다.

문항13) 철수는 골절을 1, 정상을 0으로 라벨링한 X-ray 이미지를 사용해 골절 여부를 판단하는 딥러닝(Deep Learning) 모델을 학습하려고 한다. 다양한 하이퍼파라미터(hyperparameter)에서 모델을 학습시킨 결과는 아래 [표]와 같다. 이 중 정확도(accuracy)가 최소 90% 이상인 모델을 선택하여 사용하려고 한다. 단, 골절인데 정상이라고 알려주는 경우, 환자에게 심각한 문제가 발생할 수 있으므로 이런 문제를 최소화할 수 있는 모델을 사용하려고 한다. 아래의 모델 중 가장 적합한 모델을 고르시오. [4점]

[표]

	① 모델 1	② 모델 2	③ 모델 3	④ 모델 4	⑤ 모델 5
Capacity	$5*10^6$	$7*10^9$	$8*10^6$	$6*10^5$	$3*10^7$
Precision	0.794	0.959	0.784	0.744	0.786
Recall	0.900	0.783	0.957	0.977	0.957
F1-score	0.844	0.862	0.862	0.844	0.863
Accuracy	0.900	0.925	0.908	0.892	0.909

(정답) 3

(해설) 문제에 주어진 조건에 따라 골절인데 정상이라고 판단하는 경우를 최소화하기 위해서는 [표]에 주어진 metric 중 recall 을 최대화해야 한다. 따라서 accuracy 가 90% 이상인 모델 중 recall 이 가장 큰 모델을 선택해야 한다. 단, 성능이 동일할 경우 network capacity 가 작은 모델을 선택한다.

문항14) 아래 [표]는 구글의 'AI & Machine Learning Product'에서 제공하는 API(Application Programming Interface)의 목록 일부이다. 철수가 [보기]에 주어진 애플리케이션 혹은 프로그램을 만들려고 할 때, 학습 데이터를 수집하거나 모델을 직접 학습하지 않고도 활용할 수 있는 API를 [표]에서 찾아 모두 선택한 것을 고르시오. [4점]

[표]

API	API 설명
동영상 라벨 분석	동영상의 라벨을 분석합니다.
얼굴 감지	동영상에서 얼굴을 감지합니다.
사람 감지	동영상 파일에서 사람을 감지합니다.
장면 변화 감지	동영상의 장면 변화를 분석합니다.
유해성 콘텐츠 감지	동영상의 유해성 콘텐츠를 분석합니다.
로고 인식	특정 브랜드 형태 및 로고가 있는지 감지하고 인식할 수 있습니다.
텍스트 인식	동영상에 있는 텍스트를 감지하고 스크립트를 작성합니다.
동영상의 음성 텍스트 변환	동영상의 음성을 텍스트로 변환합니다.
번역	인공신경망 기계번역 기술을 사용해 텍스트를 백 개가 넘는 언어로 즉시 번역합니다.

[보기]

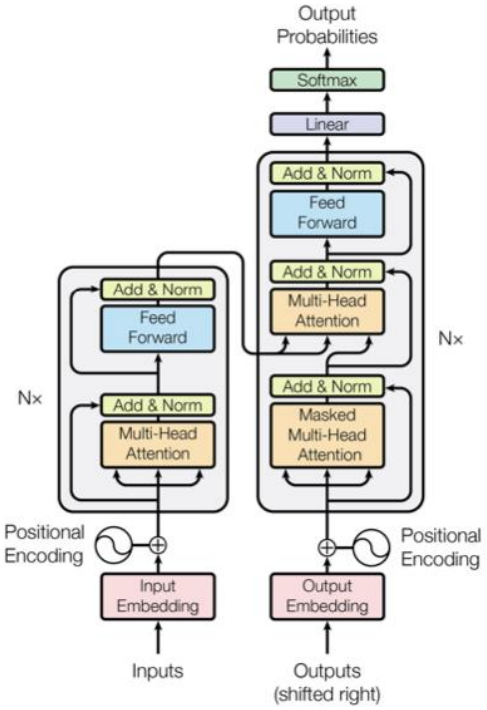
동영상 업로드 시 사전에 허가받지 않은 광고를 차단하고 외국 사용자를 위한 자막 서비스를 제공하려고 한다.
또한 초상권 보호를 위해 촬영 대상자 이외의 사람은 모두 얼굴을 가리려고 한다.

① 로고 인식, 동영상의 음성 텍스트 변환, 번역, 얼굴 감지
② 로고 인식, 텍스트 인식, 번역, 장면 변화 감지
③ 유해성 콘텐츠 감지, 텍스트 인식, 번역, 사람 감지
④ 얼굴 감지, 동영상의 음성 텍스트 변환, 번역, 장면 변화 감지
⑤ 동영상의 음성 텍스트 변환, 텍스트 인식, 로고 인식, 사람 감지

(정답) 1

(해설) ①특정 브랜드가 노출되어 광고되는 것을 방지하기 위해 로고 인식, 자막 서비스를 위해 동영상의 음성 텍스트 변환 및 번역, 얼굴을 가리는 기능을 위해 얼굴 감지가 필요하다.

문항15) 다음 Transformer 모델에 적용된 기법에 대한 설명 중 옳지 않은 것을 모두 고르시오. (3개) [4점]



[그림] Transformer 모델 구조

① Positional encoding은 입력 sequence의 특정 위치에 대한 상대적인 순서 정보(Relative positional embedding)를 입력한다.
② Multi-head attention은 입력 sequence의 순서를 head의 개수만큼 나눠서 통과시키고 합쳐준다.
③ Self-attention은 주어진 Query에 대해서 모든 Key와의 유사도를 각각 구하고, 구한 유사도를 가중치로 하여 Key와 매핑되어 있는 각각의 Values에 반영하여 입력 sequence 중 중요한 부분을 볼 수 있다.
④ Encoder-Decoder로 구성되어 있고 Decoder에서 Teacher Forcing을 사용하여 훈련된다.
⑤ Position-wise feed forward neural network는 입력되는 sequence를 모두 합쳐서 하나의 feed forward neural network를 통과하여 문맥 전체를 파악할 수 있게 한다.
⑥ Transformer 구조는 동일 레이어 수의 RNN(Recurrent Neural Network)을 이용할 때보다 연산 속도가 빠르다.

(정답) 1, 2, 5

(해설) 1-transformer 모델에서 사용한 positional encoding 은 절대적인 순서 정보(absolute positional embedding)를 입력한다. 2-multi-head attention 은 입력 sequence 의 순서를 나눈 것이 아닌 입력 차원을 head 개수로 나눠진 차원에 projection 하여 나눠서 입력하여 각 head 는 입력 sequence 를 모두 볼 수 있다. 5-position wise feed forward neural network 는 입력 sequence 각각 따로 feed forward neural network 를 통과한다.