

TensorFlow 2.0을 이용한 딥러닝 알고리즘 구현의 2가지 방법

- <https://www.tensorflow.org/overview/>

For beginners

The best place to start is with the user-friendly Sequential API. You can create models by plugging together building blocks. Run the “Hello World” example below, then visit the [tutorials](#) to learn more.

To learn ML, check out our [education page](#). Begin with curated curriculums to improve your skills in foundational ML areas.

```
import tensorflow as tf
mnist = tf.keras.datasets.mnist

(x_train, y_train), (x_test, y_test) = mnist.load_data()
x_train, x_test = x_train / 255.0, x_test / 255.0

model = tf.keras.models.Sequential([
    tf.keras.layers.Flatten(input_shape=(28, 28)),
    tf.keras.layers.Dense(128, activation='relu'),
    tf.keras.layers.Dropout(0.2),
    tf.keras.layers.Dense(10, activation='softmax')
])

model.compile(optimizer='adam',
              loss='sparse_categorical_crossentropy',
              metrics=['accuracy'])

model.fit(x_train, y_train, epochs=5)
model.evaluate(x_test, y_test)
```

For experts

The Subclassing API provides a define-by-run interface for advanced research. Create a class for your model, then write the forward pass imperatively. Easily author custom layers, activations, and training loops. Run the “Hello World” example below, then visit the [tutorials](#) to learn more.

```
class MyModel(tf.keras.Model):
    def __init__(self):
        super(MyModel, self).__init__()
        self.conv1 = Conv2D(32, 3, activation='relu')
        self.flatten = Flatten()
        self.d1 = Dense(128, activation='relu')
        self.d2 = Dense(10, activation='softmax')

    def call(self, x):
        x = self.conv1(x)
        x = self.flatten(x)
        x = self.d1(x)
        return self.d2(x)

model = MyModel()

with tf.GradientTape() as tape:
    logits = model(images)
    loss_value = loss(logits, labels)
grads = tape.gradient(loss_value, model.trainable_variables)
optimizer.apply_gradients(zip(grads, model.trainable_variables))
```

Keras Callbacks

- Keras Callbacks는 Beginners Style의 구현을 이용할 경우 사용할 수 있는 유용한 기능입니다.
- 학습과정에 필요한 다양한 기능들(TensorBoard 로그저장, Checkpoint 파일저장 등)을 손쉽게 사용할 수 있습니다.



Available Callbacks

- <https://keras.io/api/callbacks/>

Available callbacks

- Base Callback class
- ModelCheckpoint
- TensorBoard
- EarlyStopping
- LearningRateScheduler
- ReduceLROnPlateau
- RemoteMonitor
- LambdaCallback
- TerminateOnNaN
- CSVLogger
- ProgbarLogger

ModelCheckpoint

- https://keras.io/api/callbacks/model_checkpoint/

ModelCheckpoint

ModelCheckpoint class

```
tf.keras.callbacks.ModelCheckpoint(  
    filepath,  
    monitor="val_loss",  
    verbose=0,  
    save_best_only=False,  
    save_weights_only=False,  
    mode="auto",  
    save_freq="epoch",  
    options=None,  
    **kwargs  
)
```

TensorBoard

- <https://keras.io/api/callbacks/tensorboard/>

TensorBoard

TensorBoard class

```
tf.keras.callbacks.TensorBoard(  
    log_dir="logs",  
    histogram_freq=0,  
    write_graph=True,  
    write_images=False,  
    update_freq="epoch",  
    profile_batch=2,  
    embeddings_freq=0,  
    embeddings_metadata=None,  
    **kwargs  
)
```

EarlyStopping

- https://keras.io/api/callbacks/early_stopping/

EarlyStopping

EarlyStopping class

```
tf.keras.callbacks.EarlyStopping(  
    monitor="val_loss",  
    min_delta=0,  
    patience=0,  
    verbose=0,  
    mode="auto",  
    baseline=None,  
    restore_best_weights=False,  
)
```

LearningRateScheduler

- https://keras.io/api/callbacks/learning_rate_scheduler/

LearningRateScheduler

LearningRateScheduler class

```
tf.keras.callbacks.LearningRateScheduler(schedule, verbose=0)
```

CSVLogger

- https://keras.io/api/callbacks/csv_logger/

CSVLogger

CSVLogger class

```
tf.keras.callbacks.CSVLogger(filename, separator="," , append=False)
```

Keras callbacks example

- https://colab.research.google.com/drive/1KTRK61ouw5JM_WvfFIVxJS5vMgl-yGQs?usp=sharing

Callbacks in
 Keras

Thank you!
