برای این تمرین می‌خوایم با مفهوم residual بیشتر آشنا بشیم. برای این کار یک MLP با ۲۰ لایهٔ خطی پشت سر هم روی دیتاست Fashion-MNIST پیاده‌سازی کنید.

در ابتدا هیچ اتصال residual نداشته باشید. مدل را آموزش دهید و بررسی کنید که آیا مدل به‌درستی یاد می‌گیرد یا با نه. سپس نسخه‌ای از همین مدل بسازید که هر ۴ لایه یک بار از اتصال residual استفاده می‌کند. گرادیان با هم مقایسه کنید. نتایج نهایی دو مدل و سرعت یادگیری رسیدن به درصد قابل قبول رو مقایسه کنید (مثلا در چند epoch به درصد خاصی می‌رسیند)

در بخش دوم، روی یک دیتاست نامتوازن کار کنید. به‌این صورت که تعداد اعضای یکی از کلاس‌ها را به صورت رندوم تا ۸۰ درصد کاهش دهید. مدل را دوباره آموزش دهید و این بار به‌جای معیار دقت ،(Accuracy) از معیارهای Precision، Recall و F1-score برای ارزیابی استفاده کنید. ببنید می‌توانید نتایج مدل را با روش‌هایی که تا حالا یاد گرفتید بهبود دهید؟

کد کار کردن با دیتاست:

```
from torchvision import datasets, transforms
from torch.utils.data import DataLoader, Subset
import random

# Load Fashion-MNIST
transform = transforms.ToTensor()
train_set = datasets.FashionMNIST(
    root='./data', train=True, download=True, transform=transform
)

# Make a class imbalanced (reduce to 20%)
def unbalance_dataset(dataset, target_class=5, reduction_ratio(2.0=:
    indices = list(range(len(dataset)))
    targets = dataset.targets
    class_indices = [i for i in indices if targets[i] == target_class]
    other_indices = [i for i in indices if targets[i] != target_class]
    reduced_class_indices = random.sample(
        class_indices, int(len(class_indices) * reduction_ratio)
    )
    new_indices = other_indices + reduced_class_indices
    random.shuffle(new_indices)
    return Subset(dataset, new_indices)

imbalanced_train = unbalance_dataset(train_set, target_class=5,
    reduction_ratio = 2.0 )
loader = DataLoader(imbalanced_train, batch_size=128, shuffle=True)
```