Automatic Mixed Precision Training

Reference:
https://www.youtube.com/watch?v=ks3oZ7Va8HU&list=PLhhyoLH6IjfxeoooqP9rhU3HJIAVAJ3Vz&index=49
https://developer.nvidia.com/automatic-mixed-precision

Benefits of Mixed precision training

- Speeds up math-intensive operations, such as linear and convolution layers, by using Tensor Cores.
- Speeds up memory-limited operations by accessing half the bytes compared to single-precision.
- Reduces memory requirements for training models, enabling larger models or larger minibatches.

```
# Use automatic mixed precision
scaler = torch.cuda.amp.GradScaler()


# Train Network
for epoch in range(num_epochs):
    for batch_idx, (data, targets) in enumerate(tqdm(train_loader)):
        # Get data to cuda if possible
        data = data.to(device=device)
        targets = targets.to(device=device)

        # forward
        with torch.cuda.amp.autocast():
            scores = model(data)
            loss = criterion(scores, targets)

        # backward
        optimizer.zero_grad()
        scaler.scale(loss).backward()

        # gradient descent or adam step
        scaler.step(optimizer)
        scaler.update()
        # optimizer.step()
```