五种归一化

1. Batch Normalization

per channel，across，mini-batch

总结一句话BN就是：做的是通道级别的归一化（每个通道单独算），贯穿到每个mini batch，在计算统计量的时候要考虑到整个batch

```
CLASS torch.nn.BatchNorm1d(num_features, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True,        device=None, dtype=None)
```

num_features：输入的通道数

eps：用于数值的稳定性的（如分母为0）

momentum：动量，与track_running_stat联合起来用，统计量是通过滑动平均来计算，是累计的过程。作用是当走到局部最小时，此时不只是看梯度，还要根据前一步的动量

affine：默认True，做完归一化之后，还可以加一个映射，映射到一个新的分布，比如re-scale、re-center

2. Layer Normalization

per sample，per layer

总结：对单一样本计算，而且是对每一层进行计算

一般用于NLP

```
CLASS torch.nn.LayerNorm(normalized_shape, eps=1e-05, elementwise_affine=True,
device=None, dtype=None)
```

normalized_shape：需要进行归一化的shape（一般是特征维度）

eps：用于数值的稳定性的（如分母为0）

elementwise_affine：默认True，做完归一化之后，还可以加一个映射，映射到一个新的分布，比如re-scale、re-center

3. Instance Normalization

per sample，per channel

总结：对单一样本计算，还对每个通道进行计算

一般用于风格迁移

```
CLASS torch.nn.InstanceNorm1d(num_features, eps=1e-05, momentum=0.1,
affine=False,       track_running_stats=False, device=None, dtype=None)
```

num_feature：特征维度（通道维度）

eps：用于数值的稳定性的（如分母为0）

momentum：动量，与track_running_stat联合起来用，统计量是通过滑动平均来计算，是累计的过程。作用是当走到局部最小时，此时不只是看梯度，还要根据前一步的动量

affine：实例归一化中默认为False，与BN、LN不一样

4. Group Normalization

per sample，per group

与Layer Normalization很像，不过得先将Layer划分为Group

```
CLASS torch.nn.GroupNorm(num_groups, num_channels, eps=1e-05, affine=True,
device=None, dtype=None)
```

num_groups：group数目

eps：用于数值的稳定性的（如分母为0）

affine：默认True，做完归一化之后，还可以加一个映射，映射到一个新的分布，比如re-scale、re-center

5. Weight Normalization

```
torch.nn.utils.weight_norm(module, name='weight', dim=0)
```

与上述归一化方法不同的是，这里调用的不是类，而是函数，而且需要包裹一个Module