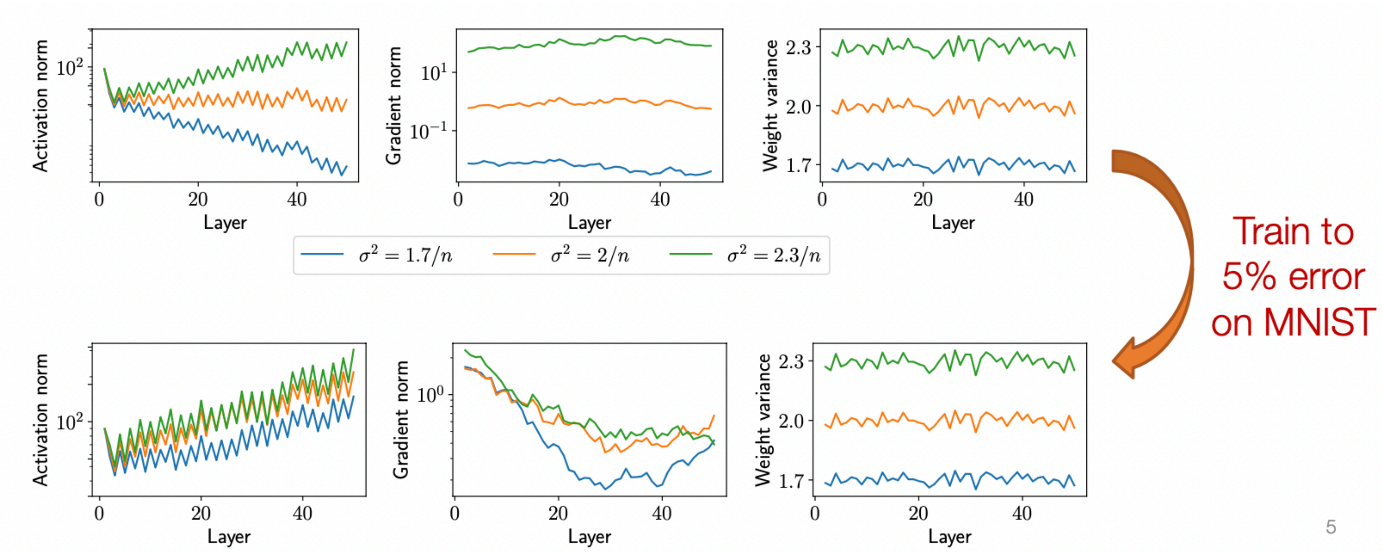
Normalization and Regularization

Normalization 和 Regularization 是加快和简化训练过程的技巧

Initialization 对激活函数和梯度的范数影响大，在训练过程中权重的变化很小



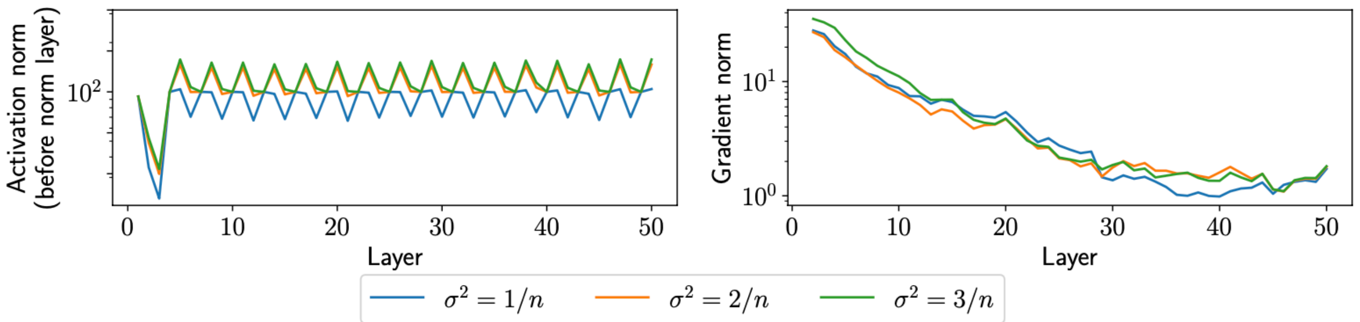
Normalization

Motivation: 加入一层 normalization 专门用于解决 initialization 带来的范数变化问题

Layer Normalization (LayerNorm)

核心思想：对每层的激活函数进行标准化（均值为 0, 方差为 1）

$$\begin{aligned} \hat{z_{i+1}} &= \sigma_i(W_i^T z_i + b_i) \\ z_{i+1} &= \frac{\hat{z_{i+1}} - E[\hat{z}_{i+1}]}{\sqrt{Var[\hat{z_{i+1}}] + \epsilon}} \end{aligned} \tag{2}$$



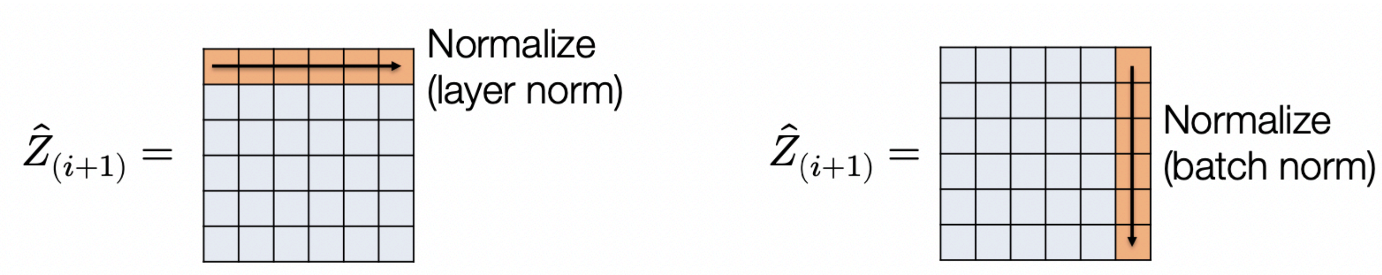
避免了激活函数/梯度在传播过程中过大的问题

在 Transformer 中很常用，但在实际的 FCN (Fully-connected Network) 中很难把 loss 降低，因为样例范数的相对值是显著的分类特征。

Batch Normalization

核心思想：在 matrix batch 中，layer norm 是对单行进行标准化，batch norm 对单列进行标准化。layer norm 在很多网络中效果不佳的原因是标准化消除了相对范数的信息，对分类造成影响。batch norm 能在执行标准化使得范数不溢出的同时确保每行的均值和方差不同。

$$\hat{Z_{i+1}} = \sigma_i(Z_i W_i + b_i^T) \tag{3}$$



Minibatch Dependence

Batch norm 的问题在于 minibatch 的样例之间存在关联，导致对某一个样例的预测结果取决于所处的 minibatch

解决方法：在测试时采用运行时的平均均值/方差

$$\begin{aligned}(z_{i+1})_j &= \frac{(\hat{z_{i+1}})_j - (\hat{\mu_{i+1}})_j}{\sqrt{(\hat{\sigma_{i+1}}^2)_j + \epsilon}} \\ \hat{\mu}_{i+1} &= \beta_1 \hat{\mu}_i + (1-\beta_1)E[\hat{z_{i+1}}] \\ \hat{\sigma_{i+1}}^2 &= \beta_2 \hat{\sigma_i}^2 + (1-\beta_2)Var[\hat{z_{i+1}}]\end{aligned} \tag{4}$$

Regularization

Overparameterized model: 参数（权重）数量比训练样例数量多。这意味着模型可以恰好拟合训练数据，会导致过拟合的发生，泛化能力弱。

Regularization 即限制函数的复杂性，以保证模型的泛化能力。

- Implicit regularization: 现有算法或框架自身对函数的限制，例如 SGD 只优化网络的一部分参数
- Explicit regularization: 显式地对网络进行修改以达到正规化的目的

L2 Regularization (Weight Decay)

核心思想：模型的复杂度可以用参数的大小来衡量，限制参数的大小意味着限制了目标函数的变化范围

因此需要在最小化 loss 的同时使参数足够小：

$$\min_{W_{1:D}} \frac{1}{m} \sum_{i=1}^{m} l(h_{W_{1:D}}(x^{(i)}), y^{(i)}) + \frac{\lambda}{2} \sum_{i=1}^{D} \|W_i\|_2^2 \tag{5}$$

梯度计算变为：

$$W_i := W_i - \alpha \nabla_{W_i} l(h(X), y) - \alpha\lambda W_i = (1-\alpha\lambda)W_i - \alpha \nabla_{W_i} l(h(X), y) \tag{6}$$

实际上就是对 W_i 乘了参数 $(1-\alpha\lambda)$, 因此也称为 Weight Decay.

通常会将 L2 regularization 实现为 optimizer 的一部分，而不是 loss function

虽然很常用，但实际并不知道参数的范数对目标函数复杂性的影响有多大

Dropout

核心思想：在训练过程中，随机将每层激活函数的一部分分量值设为 0

$$\hat{z}_{i+1} = \sigma_i(W_i^T z_i + b_i)$$
$$(z_{i+1})_j = \begin{cases} (\hat{z}_{i+1})_j/(1-p) & \text{with probability } 1-p \\ 0 & \text{with probability } p \end{cases}$$

在测试时不常用

Dropout as Stochastic Approximation

SGD 是一种 implicit regularization

可以将 dropout 类比为 SGD

$$\frac{1}{m}\sum_{i=1}^{m}\ell(h(x^{(i)}), y^{(i)}) \implies \frac{1}{|B|}\sum_{i\in B}\ell(h(x^{(i)}), y^{(i)})$$
$$z_{i+1} = \sigma_i\left(\sum_{j=1}^{n} W_{:,j}(z_i)_j\right) \implies z_{i+1} = \sigma_i\left(\frac{n}{|\mathcal{P}|}\sum_{j\in\mathcal{P}}^{n} W_{:,j}(z_i)_j\right)$$

> 因为公式对应的是 $W_i^T z_i$, 所以此处被丢弃的是 W_i 的若干行