9F 玖富
9Fbank.com
移动金融大师兄

Batch Normalization

ANN收敛比较慢的原因

- 1. 学习率不能太大
  - 对于使用mini-batch的训练方式尤为明显，因为每一个mini-batch的variance差异太大，较大的学习率会在不同的mini-batch上带来混乱
- 2. 网络权重初始值选取很重要
- 3. 激活函数的选取问题（最好使用ReLU）

Batch Normalization能帮助做什么？

- 不用小心翼翼的调整学习率
- 不用小心翼翼的初始化网络参数（权重）

- **收敛快**

- 可以*部分*代替dropout的作用

- 如何做到这些?
  - 锁定网络中的每一层的分布，使其不因为mini-batch的改变而改变

BatchNorm

- that takes a step towards reducing internal covariate shift, and in doing so dramatically accelerates the training of deep neural nets
- Batch Normalization also has a beneficial effect on the gradient flow through the network, by reducing the dependence of gradients on the scale of the parameters or of their initial values, This allows us to use much higher learning rates without the risk of divergence
- batch normalization regularizes the model and reduces the need for Dropout
- Batch Normalization makes it possible to use saturating nonlinearities by preventing the network from getting stuck in the saturated modes

启发

- LeCun, Y., Bottou, L., Orr, G., and Muller, K. Efficient backprop. In Orr, G. and K., Muller (eds.), *Neural Networks: Tricks of the trade*. Springer, 1998b

- 提到：the network training converges faster if its inputs are whitened – i.e., linearly transformed to have zero means and unit variances, and decorrelated

- 那么对每一层做这样的变化是不是更好?

如何Normalization

- Mini-Batch

- $\hat{x}^{(k)} = \frac{x^{(k)} - E[x^{(k)}]}{\sqrt{Var[x^{(k)}]}}$
  - 就是StandardScaler

- Normalization发生的位置：
  - Activation之前

- 有一个问题，这个会把输入$\hat{x}^{(k)}$限制在0均值，单位方差的区域，对于一些例如sigmoid的激活函数，这会将输入限制在“线性区域”

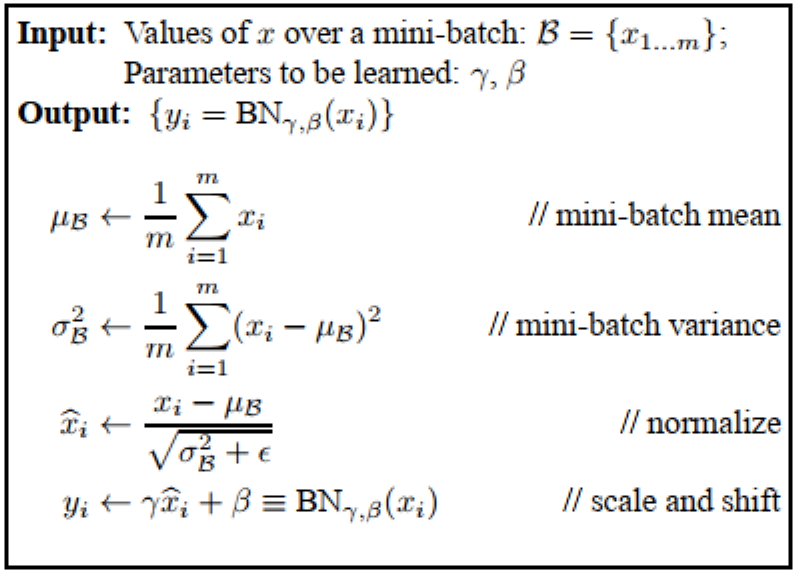
引入两个可更新参数 gamma, beta

- $\gamma^{(k)}, \beta^{(k)}$
  - 那么activation的输入变成了

$$y^{(k)} = \gamma^{(k)}\hat{x}^{(k)} + \beta^{(k)}$$

相当于把一层的某个维度的值做了一次分布转换，转换的规则（转换后的均值与方差）是可学习的值

BN Function



Input: Values of x over a mini-batch: $\mathcal{B} = \{x_{1\dots m}\}$;
Parameters to be learned: γ, β
Output: $\{y_i = \mathrm{BN}_{\gamma,\beta}(x_i)\}$

$$\mu_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i \qquad \text{// mini-batch mean}$$

$$\sigma_{\mathcal{B}}^2 \leftarrow \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_{\mathcal{B}})^2 \qquad \text{// mini-batch variance}$$

$$\widehat{x}_i \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}} \qquad \text{// normalize}$$

$$y_i \leftarrow \gamma\widehat{x}_i + \beta \equiv \mathrm{BN}_{\gamma,\beta}(x_i) \qquad \text{// scale and shift}$$

Algorithm 1: Batch Normalizing Transform, applied to activation x over a mini-batch.

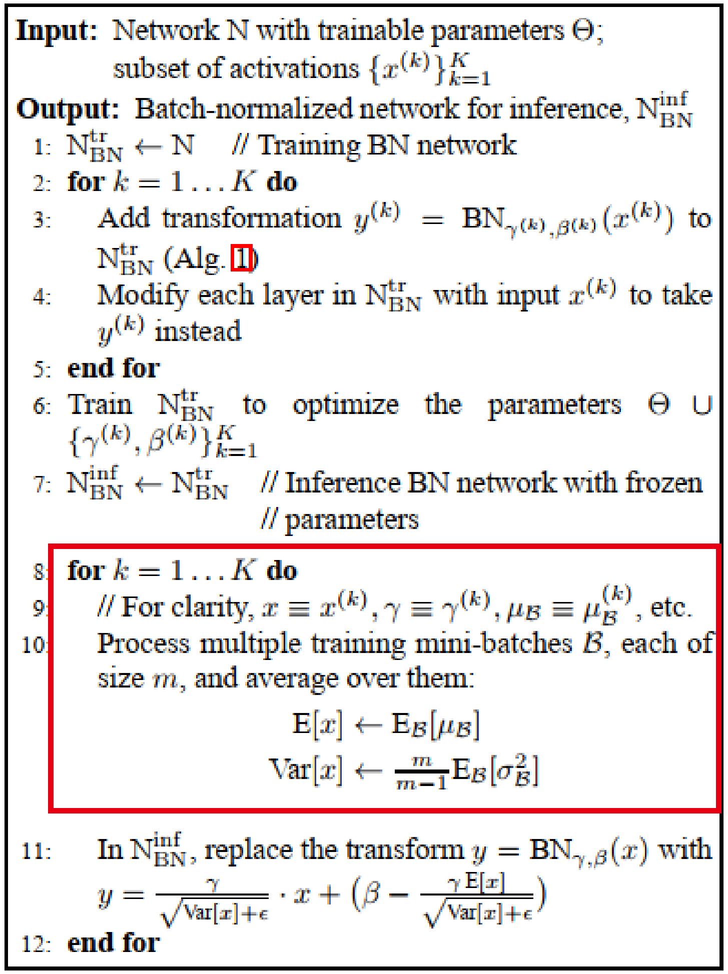
BP

- 因为添加的 BN function是连续函数，完全是可微的，所以仍然可以使用链式法则求解一阶导数（梯度）

对于单个的实体的预测（inference）如何解决

- Mini-batch的训练可以通过求解Mean和Variance的办法来求解，但是在预测一个实体的时候，是没有mean和variance的。
- 解决办法是
  - 在训练的过程中求解各个batch的平均值的平均值，求解batch的unbiased平均值

$$\mathrm{Var}[x] \leftarrow \frac{m}{m-1}\mathrm{E}_{\mathcal{B}}[\sigma_{\mathcal{B}}^2]$$



```
Input: Network N with trainable parameters $\Theta$;
       subset of activations $\{x^{(k)}\}_{k=1}^K$
Output: Batch-normalized network for inference, $\mathrm{N}_{\mathrm{BN}}^{\mathrm{inf}}$
1:  $\mathrm{N}_{\mathrm{BN}}^{\mathrm{tr}} \leftarrow \mathrm{N}$   // Training BN network
2:  for $k = 1 \ldots K$ do
3:      Add transformation $y^{(k)} = \mathrm{BN}_{\gamma^{(k)},\beta^{(k)}}(x^{(k)})$ to
        $\mathrm{N}_{\mathrm{BN}}^{\mathrm{tr}}$ (Alg. 1)
4:      Modify each layer in $\mathrm{N}_{\mathrm{BN}}^{\mathrm{tr}}$ with input $x^{(k)}$ to take
        $y^{(k)}$ instead
5:  end for
6:  Train $\mathrm{N}_{\mathrm{BN}}^{\mathrm{tr}}$ to optimize the parameters $\Theta \cup$
    $\{\gamma^{(k)}, \beta^{(k)}\}_{k=1}^K$
7:  $\mathrm{N}_{\mathrm{BN}}^{\mathrm{inf}} \leftarrow \mathrm{N}_{\mathrm{BN}}^{\mathrm{tr}}$   // Inference BN network with frozen
                                   // parameters
8:  for $k = 1 \ldots K$ do
9:      // For clarity, $x \equiv x^{(k)}, \gamma \equiv \gamma^{(k)}, \mu_{\mathcal{B}} \equiv \mu_{\mathcal{B}}^{(k)}$, etc.
10:     Process multiple training mini-batches $\mathcal{B}$, each of
        size $m$, and average over them:
            $\mathrm{E}[x] \leftarrow \mathrm{E}_{\mathcal{B}}[\mu_{\mathcal{B}}]$
            $\mathrm{Var}[x] \leftarrow \frac{m}{m-1}\mathrm{E}_{\mathcal{B}}[\sigma_{\mathcal{B}}^2]$
11:     In $\mathrm{N}_{\mathrm{BN}}^{\mathrm{inf}}$, replace the transform $y = \mathrm{BN}_{\gamma,\beta}(x)$ with
        $y = \frac{\gamma}{\sqrt{\mathrm{Var}[x]+\epsilon}} \cdot x + \left(\beta - \frac{\gamma\,\mathrm{E}[x]}{\sqrt{\mathrm{Var}[x]+\epsilon}}\right)$
12: end for
```

Algorithm 2: Training a Batch-Normalized Network

但是！。。。

- 为什么BatchNorm会有作用?
  - 是因为BatchNorm改变了layer输入的分布?
  - 为什么改变了这个输入分布就会有作用呢?

- 请看How Does Batch Normalization Help Optimization? (No, It Is Not About Internal Covariate Shift)

  - The popular belief is that this effectiveness stems from controlling the change of the layers’ input distributions during training to reduce the so-called “internal covariate shift”.
  - In this work, we demonstrate that such distributional stability of layer inputs has little to do with the success of BatchNorm
  - Insstead, it makes the optimization landscape significantly smoother. This smoothness induces a more predictive and stable behavior of the gradients, allowing for faster training

延伸阅读

- https://arxiv.org/pdf/1502.03167.pdf
- https://arxiv.org/pdf/1805.11604.pdf
- https://www.youtube.com/watch?v=tNIpEZLv_eg&list=PLkDaE6sCZn6Hn0vK8co82zjQtt3T2Nkqc&index=27