Batch Normalization

- Normalizes layer inputs
  - Mean = 0
  - Variance = 1
- Then, puts these normalized inputs through a linear function
  - $\Gamma x + \beta$
  - Allows the net to “denormalize” the input, if it finds it beneficial to do so
- Reduces internal covariate shift
  - Reduces dependency that later layers have on earlier layers, so changes in earlier layers won’t affect later layers that much
    - Speeds up training
  - Internal Covariate Shift =
    - Change in distribution of network activations due to change in network parameters during training
    - “when your inputs change on you, and your algorithm can’t deal with it”
- Lets us use higher learning rates
- Reduces need for dropout
- Slightly regularizes model
- Makes it possible to use saturating nonlinearities
- **Input:** Values of x over a mini-batch: $\mathcal{B} = \{x_{1...m}\}$;
  Parameters to be learned: γ, β

  Output: $\{y_i = \mathrm{BN}_{\gamma,\beta}(x_i)\}$

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

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

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

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