

Map-reduce

Batch gradient descent:

$$m = 400$$

$$m = 400,000,000$$

$$\theta_j := \theta_j - \alpha \frac{1}{400} \sum_{i=1}^{400} (h_{\theta}(x^{(i)}) - y^{(i)}) x_j^{(i)}$$

Machine 1: Use $(x^{(1)}, y^{(1)}), \dots, (x^{(100)}, y^{(100)})$.

$$\text{temp}_j^{(1)} = \sum_{i=1}^{100} (h_{\theta}(x^{(i)}) - y^{(i)}) \cdot x_j^{(i)}$$

Machine 2: Use $(x^{(101)}, y^{(101)}), \dots, (x^{(200)}, y^{(200)})$.

$$\text{temp}_j^{(2)} = \sum_{i=101}^{200} (h_{\theta}(x^{(i)}) - y^{(i)}) \cdot x_j^{(i)}$$

Machine 3: Use $(x^{(201)}, y^{(201)}), \dots, (x^{(300)}, y^{(300)})$.

$$\text{temp}_j^{(3)} = \sum_{i=201}^{300} (h_{\theta}(x^{(i)}) - y^{(i)}) \cdot x_j^{(i)}$$

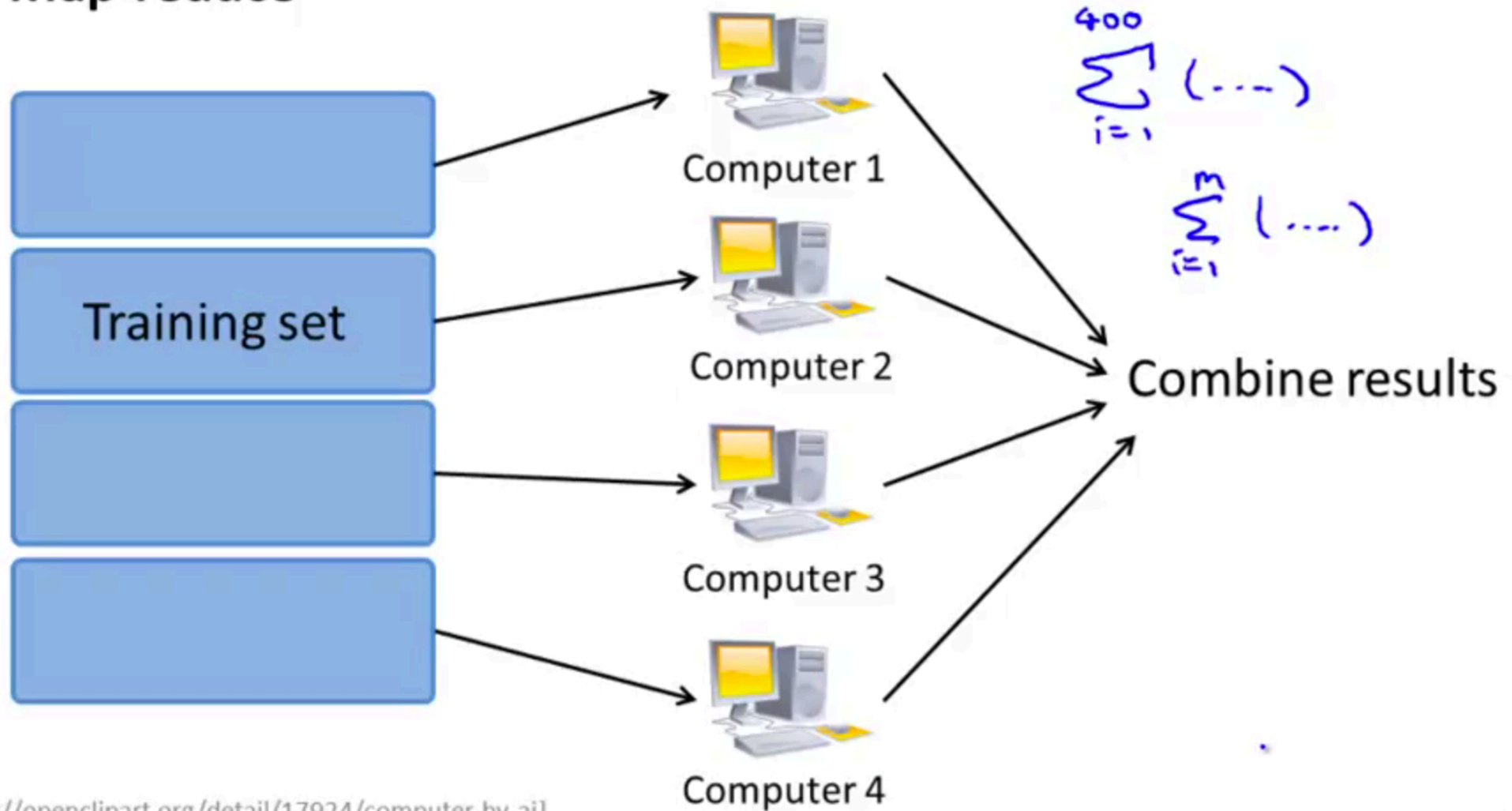
Machine 4: Use $(x^{(301)}, y^{(301)}), \dots, (x^{(400)}, y^{(400)})$.

$$\text{temp}_j^{(4)} = \sum_{i=301}^{400} (h_{\theta}(x^{(i)}) - y^{(i)}) \cdot x_j^{(i)}$$

Combine:

$$\begin{aligned} \theta_j := & \theta_j \\ & - \alpha \frac{1}{400} (\\ & \text{temp}_j^{(1)} + \text{temp}_j^{(2)} \\ & + \text{temp}_j^{(3)} + \text{temp}_j^{(4)}) \\ & (j = 0, \dots, n) \end{aligned}$$

Map-reduce



Map-reduce and summation over the training set

Many learning algorithms can be expressed as computing sums of functions over the training set.

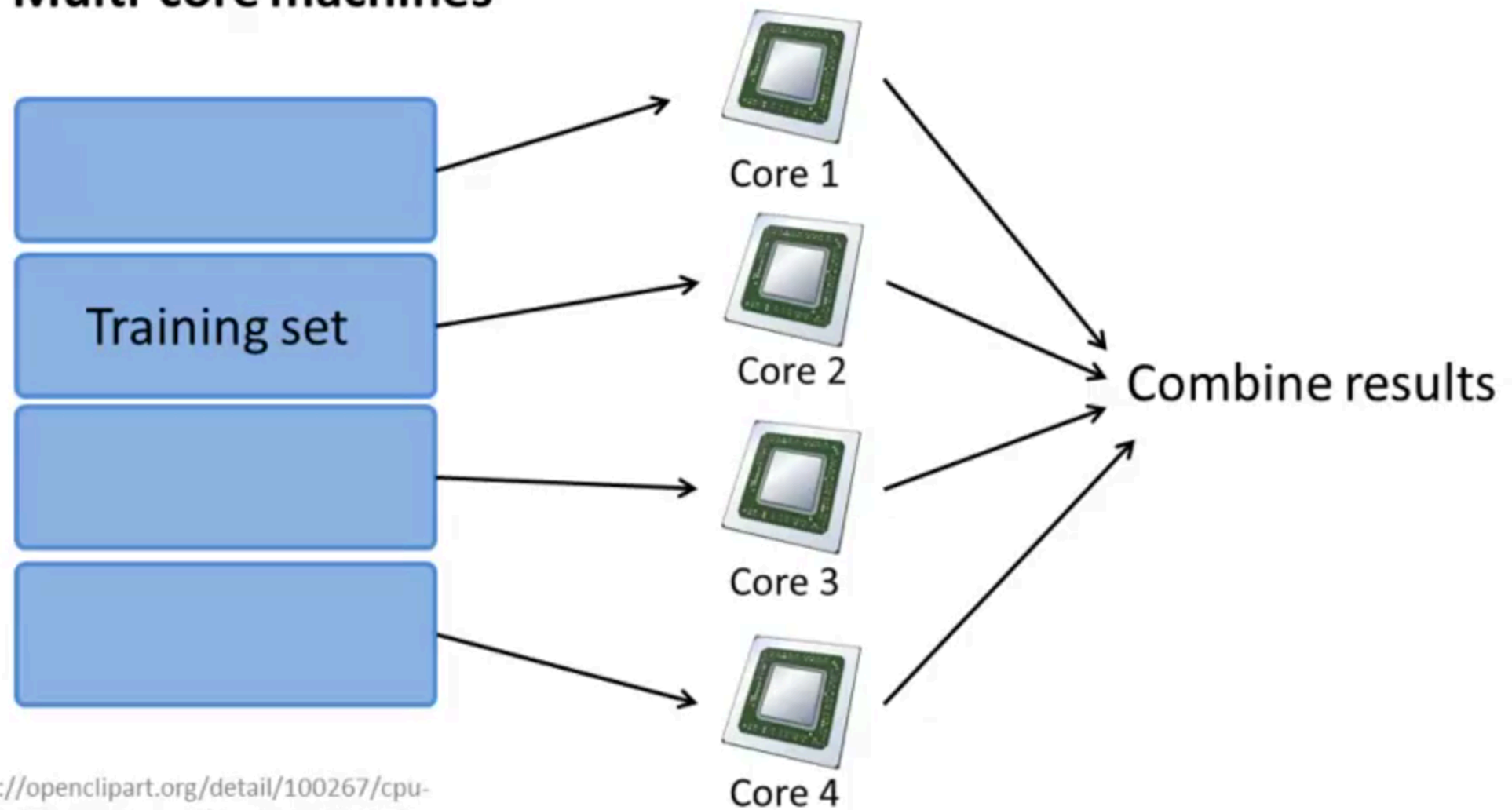
E.g. for advanced optimization, with logistic regression, need:

$$\rightarrow \underline{J_{train}(\theta)} = -\frac{1}{m} \sum_{i=1}^m \underline{y^{(i)} \log h_{\theta}(x^{(i)}) - (1 - y^{(i)}) \log(1 - h_{\theta}(x^{(i)}))}$$

$$\rightarrow \underline{\frac{\partial}{\partial \theta_j} J_{train}(\theta)} = \frac{1}{m} \sum_{i=1}^m \underline{(h_{\theta}(x^{(i)}) - y^{(i)}) \cdot x_j^{(i)}}$$

$temp^{(i)}$ $temp_j^{(i)} \leftarrow$

Multi-core machines



[[http://openclipart.org/detail/100267/cpu-\(central-processing-unit\)-by-ivak-100267](http://openclipart.org/detail/100267/cpu-(central-processing-unit)-by-ivak-100267)]