(Network) Efficient Large-Scale Distributed Training of Conditional Maximum Entropy Models

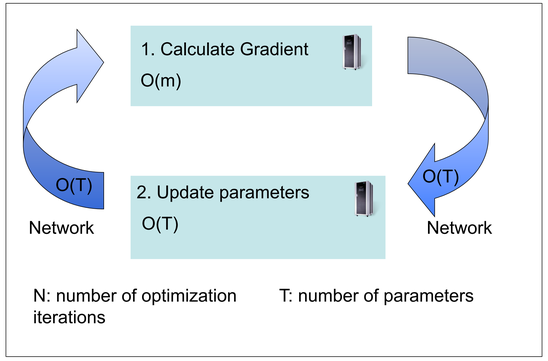
Gideon Mann, Ryan McDonald, Mehryar Mohri, Nathan Silberman, Daniel Walker

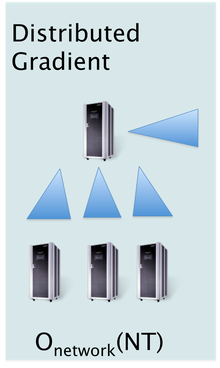
Google

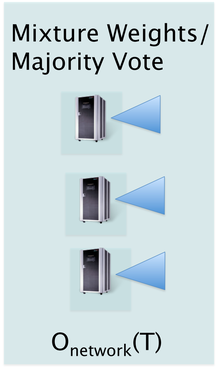
Training Maximum Entropy Models for very large data sets is challenging

$$w = \text{argmin } F(w) = \text{argmin } \lambda||w||^2 - \frac{1}{m}\sum_i \log p(y^i|x^i)$$

w : weights
m : training set size.
λ : regularization hyper-parameter.







Mixture weights are more (network) efficient, but

(1) what do they converge to?

(2) how fast do they converge?

Analysis:

(1) Convergence to a unique value at a rate of $\frac{1}{\sqrt{m}}$

$$w^\star = \text{argmin } \lambda||w||^2 - E_{z\sim D}[L_z(w)]$$

$$||w - w^\star|| \leq \frac{R}{\lambda\sqrt{m/2}}(1 + \sqrt{\log\ 1/\delta})$$

(2) Weight mixtures converge at a similar rate

$$||w_u - w^\star|| \leq \frac{R}{\lambda\sqrt{m/2}}(1 + \sqrt{\log\ 1/\delta}) + E[||w_u - w^\star||]$$

$\mathbf{w_u}$: is the average of a set of independently estimated models

Conclusion: Mixture weights converge to the same point and at a comparable rate as distributed gradient with significantly reduced network use.

Related Work

- [Chu et al. 2007] demonstrates how to perform distributed maxent and other algorithms with MapReduce.
- [Zhang 2004] and others look at stochastic gradient methods as a means to speed training time.
- [Tsitsiklis 1986] provides convergence bounds for asychronous gradient-based optimization.
- [Collins et al. 2002] propose an alternative parallel training algorithm based on connections to boosting.

Experiments with 1M to 1B instances

		Accuracy	Wall Clock	Cumulative CPU	Network Usage
English POS p = 10 pm = 1M |Y| = 24 |X| = 500k 0.001 sparsity	Distributed gradient	97.60%	17.5 m	11.0 h	652 GB
	Mixture Weight	96.80%	5 m	11.5 h	0.015 GB
Sentiment p = 10 pm = 9M |Y| = 3 |X| = 500k 0.001 sparsity	Distributed Gradient	81.18%	104 m	123 h	367 GB
	Mixture Weight	81.30%	110 m	163 h	9 GB
RCV1-v2 p = 10 pm = 26M |Y| = 103 |X| = 10k Sparsity = 0.08	Distributed gradient	27.03%	48 m	407 h	479 GB
	Mixture Weight	27.15%	56 m	473 h	0.108 GB
Speech p = 499 pm = 50M |Y| = 129 |X| = 39 Sparsity = 1.0	Distributed Gradient	34.95%	160 m	511 h	200 GB
	Mixture Weight	34.99%	130 m	534 h	158 GB
Deja News p = 200 pm = 306 M |Y| = 8 |X| = 50k Sparsity = 0.002	Distributed Gradient	64.74%	327 m	733 h	5,283 GB
	Mixture Weight	65.46%	316 m	707 h	48 GB
Deja News p = 200 pm = 306 M |Y| = 8 |X| = 250k Sparsity = 0.0004	Distributed Gradient	67.03%	340 m	698 h	17,428 GB
	Mixture Weight	66.86%	300 m	710 h	65 GB
Gigaword unigram prediction p = 1k pm = 1B |Y| = 96 |X| = 10k Sparsity = 0.001	Distributed Gradient	51.16%	240 m	18,598 h	13,000 GB
	Mixture Weight	50.12%	215 m	17998 h	21 GB

Mixture weights always have less network usage, usually less wall clock time (5/7), often the best accuracy (4/7), and for the very large data sizes the least overall cumulative CPU usage.