

Методы стохастической оптимизации нейронных сетей

Практикум на ЭВМ 2017/2018

Солоткий Михаил

МГУ имени М. В. Ломоносова, факультет ВМК, кафедра ММП

15 марта 2018 года

Актуальность

- Современные нейронные сети могут состоять из более чем 100 слоёв, иметь порядка 10^7 параметров (ResNet-152: 152 слоя, 60.2М параметров).
- Нужны быстрые методы оптимизации нейронных сетей, учитывающие специфику данных.
- Большие данные бывают разреженными.

Модификации градиентного спуска

- Стохастический градиентный спуск (SGD):

$$\theta_t = \theta_{t-1} - \eta_t \nabla_{\theta} J(\theta_{t-1}; x^{(i)}; y^{(i)})$$

- + гарантируется сходимость для выпуклых $C^{1,1}(\mathcal{H})$ функций при $\eta_t \rightarrow 0$, $\sum_{i=1}^{\infty} \eta_t = \infty$, $\sum_{i=1}^{\infty} \eta_t^2 < \infty$;
 - + онлайновый, подходит для больших данных;
 - нестабильная сходимость.
- Мини-пакетный (mini-batch) градиентный спуск:

$$\theta_t = \theta_{t-1} - \eta_t \nabla_{\theta} J(\theta_{t-1}; x^{(i;i+k)}; y^{(i;i+k)})$$

- + уменьшает разброс значений параметра, более стабильная сходимость.

Недостатки SGD и mini-batch GD

- Нейронные сети - часто невыпуклые функции с большим числом локальных оптимумов и стационарных точек.
- SGD может застревать в седловых точках, так как градиент близок к нулю во всех направлениях.
- Сложно выбирать длину шага, чтобы метод быстро сходился. Возможны осцилляции вокруг точки минимума.
- Запланированное уменьшение длины шага - не универсальный метод
- Одинаковая длина шага для всех координат.

Примеры улучшений

- Momentum:

$$v_t = \gamma v_{t-1} + \eta \nabla_{\theta} J(\theta_{t-1})$$

$$\theta_t = \theta_{t-1} - v_t$$

- $\gamma \in [0, 1)$, обычно берут $\gamma = 0.9$;
- + уменьшает осцилляцию и ускоряет движение в правильном направлении;
- иногда пропускает локальный минимум.



Рис.: SGD без momentum



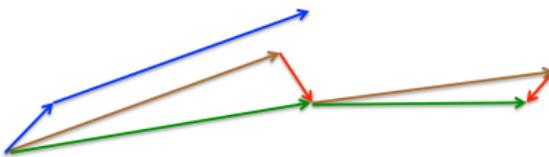
Рис.: SGD + momentum

Примеры улучшений

- Быстрый градиентный метод Нестерова:

$$v_t = \gamma v_{t-1} + \eta \nabla_{\theta} J(\theta_{t-1} - \gamma v_{t-1})$$

$$\theta_t = \theta_{t-1} - v_t$$



- + скорость сходимости $O(\frac{1}{k^2})$ для выпуклых $C^{1,1}(\mathcal{H})$ функций в отличие от классического градиентного спуска со скоростью $O(\frac{1}{k})$;
- + уменьшает шанс пропустить локальный оптимум.

Адаптивные методы

- AdaGrad:

$$g_{t,i} = \nabla_{\theta_i} J(\theta_{t,i}) \quad G_{t,ii} = \sum_{k=1}^t (g_{k,i})^2$$
$$\theta_{t+1,i} = \theta_{t,i} - \frac{\eta}{\sqrt{G_{t,ii} + \varepsilon}} \cdot g_{t,i}$$

- обычно берут $\eta = 10^{-2}$;
- + автоматически настраиваемая длина шага для каждой координаты;
- + подходит для обучения на разреженных данных;
- длина шага быстро убывает так, что градиентный спуск останавливается.

- Примеры задач:

- Обучение GloVe word embedding'ов;
- Построение рекомендательной системы.

John Duchi, Elad Hazan, and Yoram Singer. Adaptive Subgradient Methods for Online Learning and Stochastic Optimization. Journal of Machine Learning Research, 12:2121–2159, 2011.

Адаптивные методы

- RMSprop:

$$g_{t,i} = \nabla_{\theta_i} J(\theta_{t,i})$$

$$E[g^2]_t = \gamma E[g^2]_{t-1} + (1 - \gamma) g_t^2$$

$$\theta_{t+1,i} = \theta_{t,i} - \frac{\eta}{\sqrt{E[g^2]_{t,i} + \varepsilon}} \cdot g_{t,i}$$

$$\theta_{t+1} = \theta_t - \frac{\eta}{RMS[g]_t} \odot g_t$$

- рекомендация (от Geoffrey Hinton) брать $\eta = 10^{-3}$, $\gamma = 0.9$;
 - + нет остановки спуска, как в AdaGrad.

- AdaDelta:

$$\theta_{t+1} = \theta_t + \Delta\theta_t$$

$$\theta_{t+1} = \theta_t - \frac{RMS[\Delta\theta_{t-1}]}{RMS[g]_t} \odot g_t$$

Matthew D. Zeiler. ADADELTA: An Adaptive Learning Rate Method. arXiv preprint
arXiv:1212.5701, 2012.

Адаптивные методы

- Adam (Adaptive Moment Estimation):

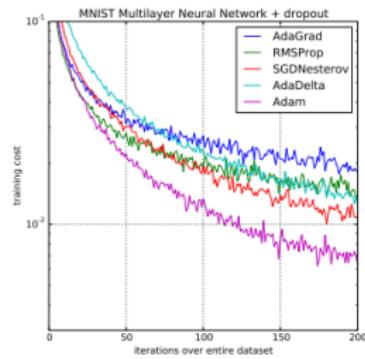
$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t \quad \hat{m}_t = \frac{m_t}{1 - \beta_1^t}$$

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2 \quad \hat{v}_t = \frac{v_t}{1 - \beta_2^t}$$

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\hat{v}_t} + \varepsilon} \hat{m}_t$$

- рекомендации от авторов:
 $\eta = 2 \cdot 10^{-3}, \beta_1 = 0.9, \beta_2 = 0.999, \varepsilon = 10^{-8}$;
- + эксперименты показывают, что Adam обходит все предыдущие методы по скорости (достаточное число итераций);
- + Вблизи оптимума Adam работает немного быстрее, чем RMSprop за счёт использования momentum.

Сопоставление методов



Рекомендации

Какой метод использовать для обучения нейронной сети:

- + Adam, возможно, лучший выбор.
- + AdaDelta, RMSprop тоже хороши.
- AdaGrad рано останавливается.
- SGD, Nesterov, momentum:
 - плохо работают с разреженными данными;
 - необходим подбор длины шага.

Что лучше заработает в конкретной ситуации - неизвестно.