尊敬的吴老师，您好

上上周的时候我联系到了论文作者，得到了数据集。我将训练过程用C++实现了一下。下面我先说一下对投影梯度下降的理解，以及遇到的一些问题。

论文的公式(8)

$$\text{Ł}(C) = -\sum_{v\in V}\sum_{D_{v,i}\in\Gamma(v)}^{N}(n_{z_{v,i},D_{v,i}} log\ p(z_{v,i} \mid z_{v,i-1}, D_{v,i}, \delta)) + \gamma_I \left\|I\right\|_F^2 + \gamma_S \left\|S\right\|_F^2$$
$$s.t.\ I_{ij} \geq 0, S_{ij} \geq 0, \forall i, j$$

是一个约束优化问题。我们先考虑约束优化问题的标准形式

$$\begin{aligned} &min_{x\in R^n} \quad f(x) \\ &subject\ to \quad l_i \leq x_i \leq u_i, \quad i = 1, \ldots, n, \end{aligned}$$

其中$f(x) : R_n \to R$是连续可微函数，l和u分别是约束的下界和上界。投影梯度算法通过下面的式子来更新x^k得到x^{k+1}:

$$x^{k+1} = P\left[x^k - \alpha^k \nabla f(x^k)\right]$$

其中

$$P\left[x_i\right] = \begin{cases} x_i & if \quad l_i < x_i < u_i \\ u_i & if \quad x_i \geq u_i \\ l_i & if \quad x_i \leq l_i \end{cases}$$

将x_i投影到了可行域。不同的梯度投影算法的区别在于步长α^k的选择方式。最常用的梯度投影算法是“Armilo rule along the projection arc”。如算法(1)所示

Algorithm 1 Projected gradient for bound-constrained optimization

[] Given $0 < \beta < 1, 0 < \sigma < 1$. Initialize any feasible x^1.

For $k = 1, 2, \ldots$

$$x^{k+1} = P\left[x^k - \alpha^k \nabla f(x^k)\right]$$

where $\alpha_k = \beta^{t_k}$, and t_k is the first non-negative integer t for which

$$f(x^{k+1}) - f(x^k) \leq \sigma \nabla f(x^k)^T (x^{k+1} - x^k.$$

其中公式

$$f(x^{k+1}) - f(x^k) \leq \sigma \nabla f(x^k)^T (x^{k+1} - x^k \tag{1}$$

保证了每次迭代下降的值足够大。通过不断尝试$1, \beta, \beta^2, \ldots$,得到$\alpha_k$。通常情况下，$\sigma$的取值是0.01，$\beta = 0.1$。寻找合适的$\alpha_k$是Algorithm (1)中最耗时的操作。考虑到α_{k-1}和α_k可能是相同的，有人提出了Algorithm (2)。

Algorithm 2 An improved projected gradient method

[] Given $0 < \beta < 1, 0 < \sigma < 1$. Initialize any feasible x^1. Set $\alpha_0 = 1$

For $k = 1, 2, \ldots$

Assign $\alpha_k \leftarrow \alpha_{k-1}/\beta$

If α_k satisfies (1), repeatedly increase it by

$$\alpha_k \leftarrow \alpha_k/\beta$$

until either α_k satisfies (1) or $x(\alpha_k/\beta) = x(\alpha_k)$

Else repeatedly decrease α_k by

$$\alpha_k \leftarrow \alpha_k \cdot \beta$$

until α_k satisfies (1)

Set

$$x^{k+1} = P\left[x^k - \alpha^k \nabla f(x^k)\right]$$

回到论文中的公式(8)，由于I和S只有下界约束，因此Algorithm (2)中的P应为

$$P\left[x_i\right] = \begin{cases} x_i & if \quad 0 \le x_i \\ 0 & if \quad x_i < 0 \end{cases}$$

训练情况

我选取了作者提供的training-data-1做训练，共计395832条信息链。每一次迭代下降需要耗时5s，跑了四个多小时，代价值L(C)从8.6×10^6下降到了7.7×10^6。在后面的迭代下降过程中，只有几十甚至个位数的减小。

对于预测部分，作者没有详细解释，我还在摸索中。

学生王超民，2016年5月25日