Stochastic Universal Gradient

Evgenii Lagutin
Optimization Class Project. MIPT

Introduction

The universal gradient method is known to be a good approach to numerical optimization when one doesn't have information about the Lipsitz constant of the gradient. This adaptive method adjusts L at each step of the optimization process and holds the following estimation of the number of calls to the oracle, returning the gradient of the minimized function:

$$N = \inf_{\nu \in [0,1]} \left(\frac{2L_\nu R^{1+\nu}}{\varepsilon} \right)^{\frac{2}{1+\nu}},$$

$$\|\nabla f(x) - \nabla f(y)\|_2 \le L_\nu \|y-x\|_2^\nu, \nu \in [0,1],\ L_0 < \infty$$

But this estimation hasn't been transferred on the stochastic case. The purpose of the project is to investigate the effectiveness of the stochastic universal gradient method in practice.

Algorithm

Adaptive Stochastic Gradient (Spokoiny's practical variant)

Input: lower estimate for the variance of the gradient $D_0 \le D$, accuracy $0 < \varepsilon < \frac{D_0}{L}$, starting point $x_0 \in Q$, initial guess $L_{-1} > 0$

for $k = 0, 1, \ldots$ **do**
  Set $i_k = 0$. Set $r^k = \lceil \frac{2D_0}{L_{k-1}} \varepsilon \rceil$, generate i.i.d. $\xi_K^i,\ i = 1, \ldots, r^k$
  repeat
    Set $L_k = 2^{i_k - 1} L_{k-1}$
    Calculate $\tilde{g}(x_k) = \frac{1}{r^k} \sum_{i=1}^{r^k} \nabla f(x_k, \xi_k^i)$.
    Calculate $w_k = x_k - \frac{1}{2L_k} \tilde{g}(x_k)$.
    Calculate $\tilde{f}(x_k) = \frac{1}{r_k} \sum_{i=1}^{r^k} f(x_k, \xi_k^i)$ and $\tilde{f}(w_k) = \frac{1}{r^k} \sum_{i=1}^{r^k} f(w_k, \xi_k^i)$.
    Set $i_k = i_k + 1$.
  until
    $\tilde{f}(w_k) \le \tilde{f}(x_k) + \langle \tilde{g}(x_k), w_k - x_k \rangle + \frac{2L_k}{2} \|w_k - x_k\|_2^2 + \frac{\varepsilon}{10}$.
  Set $x_{k+1} = w_k,\ k = k+1$.
end for

Optimization of deep neural networks

Let $g(x)$ be a stochastic gradient of the function being minimized.
In every iteration we have to check if the following inequality is satisfied:

$$f(w) \le f(x) + \langle g(x), w - x \rangle + \frac{2L}{2} \|w - x\|_2^2 + \frac{\varepsilon}{10}$$

Substituting w with the its definition expression, $w = x - \frac{1}{2L} g(x)$
We will get $f(w) \le f(x) - \frac{1}{2L} \|g(x)\|_2^2 + \frac{2L}{2} \frac{1}{4L^2} \|g(x)\|_2^2 + \frac{\varepsilon}{10}$
or $f(w) \le f(x) - \frac{1}{4L} \|g(x)\|_2^2 + \frac{\varepsilon}{10}$

Consider f(x) to be a function of a range of matrices and vectors:

$$f(x) = f(W_1, b_1, \ldots, W_n, b_n),$$

$$df(W_1, b_1, \ldots, W_n, b_n) = \sum_{i=1}^{n} \left(\frac{\partial f}{\partial W_i} dW_i + \frac{\partial f}{\partial b_i} db_i \right)(W_1, b_1, \ldots, W_n, b_n)$$

The goal is to represent df in this fashion:

$$df(x) = \langle g(x), dx \rangle$$

In this case $g(x)$ is the gradient.
Let's notice that in case of x is vector, $x \in \mathbb{R}^n$, $g(x) \in \mathbb{R}^n$

$$\langle g(x), x \rangle = \sum_{i=1}^{n} g_i(x) x_i$$

and so we do if X is a matrix: $X \in Mat(n \times m)$, $g(X) \in Mat(n \times m)$

$$\langle g(X), X \rangle = \mathbf{tr} g(X) X = g(X) \cdot X = \textstyle\sum_{i=1}^{n} \sum_{j=1}^{m} g_{ij}(X) X_{ij}$$

That means we may consider X as a vector $(x_{11}, x_{12}, \ldots, x_{1m}, x_{21}, \ldots, x_{nm})$ of dimension nm and the result will not change.

Such reasoning allows us to compute the second norm of the gradient in a following way:

$$\|g(x)\|_2^2 = \|g(W_1, b_1, \ldots, W_n, b_n)\|_2^2 = \sum_{i=1}^{n} (g_{W_1}(x) \cdot g_{W_1}(x) + \langle g_{b_1}(x), g_{b_1}(x) \rangle)$$

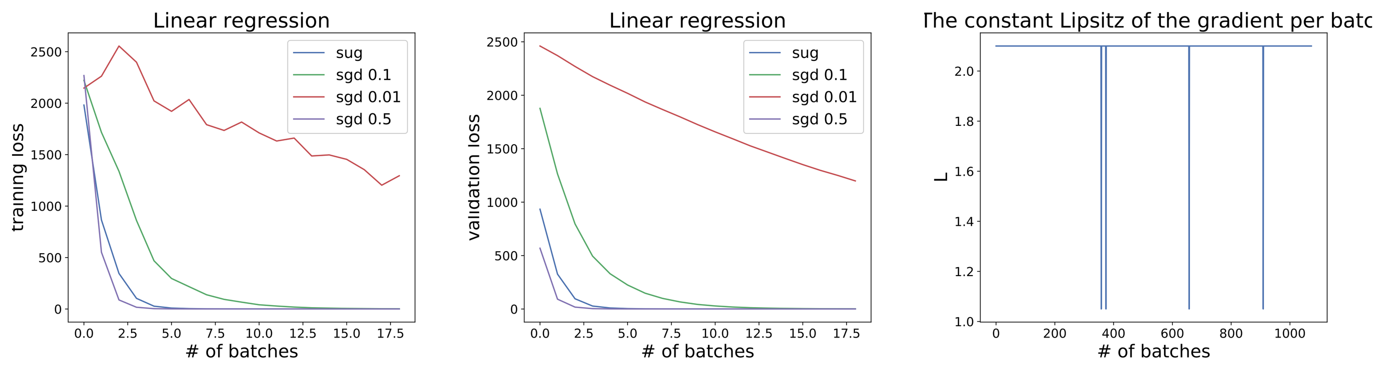
Numerical Experiments

Linear Regression

$$x_i \sim \mathcal{N}(0, I),\quad i = 1..n,\quad I \in \mathbb{R}^{m^2}$$
$$y_i = \theta^T x_i + \varepsilon,\quad \varepsilon \sim \mathcal{N}(0, \sigma^2)$$
$$y = X\theta + \varepsilon,\quad \varepsilon \sim \mathcal{N}(0, \Sigma)$$

$$L(\theta, X) = \frac{1}{m} \sum_{i=1}^{m} |x_i \theta - y_i|^2 = \frac{1}{m} \|X\theta - y\|_2^2$$

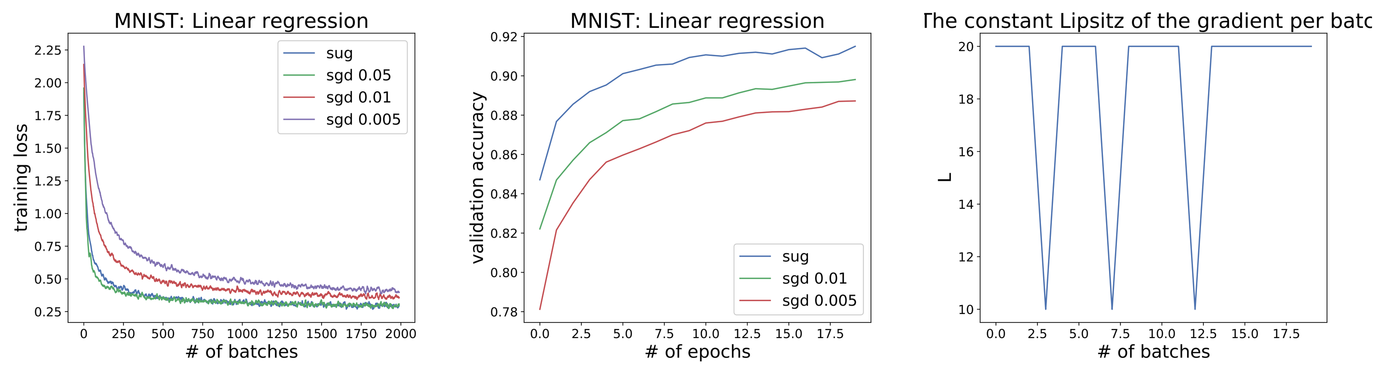
It is easy to show that L is equal to $\lambda_{max} \left(\sum_{i=1}^{m} x_i^T x_i \right)$. In this experiment it was ≈ 2.1



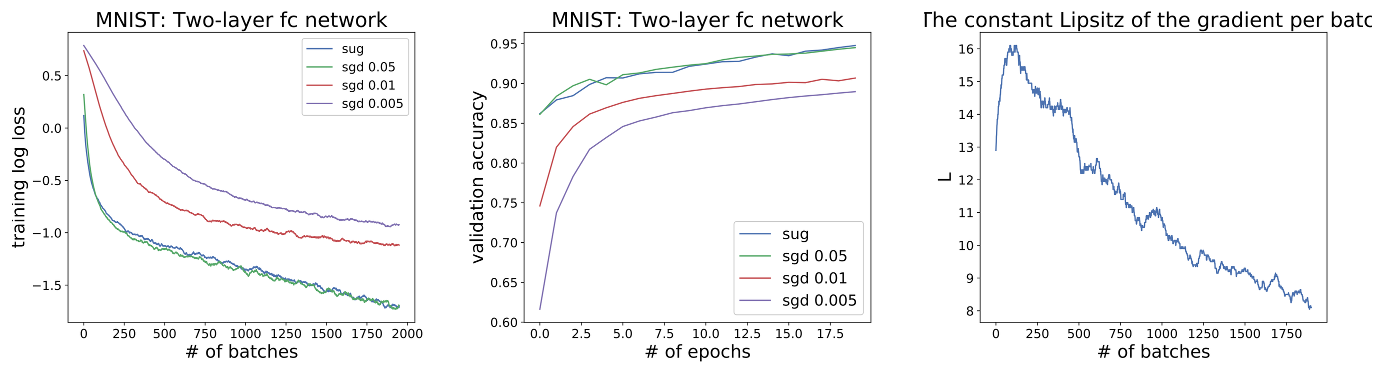
MNIST

The models selected for this experiment are: logistic regression, 2-layer fully-connected NN, simple convolutional NN.

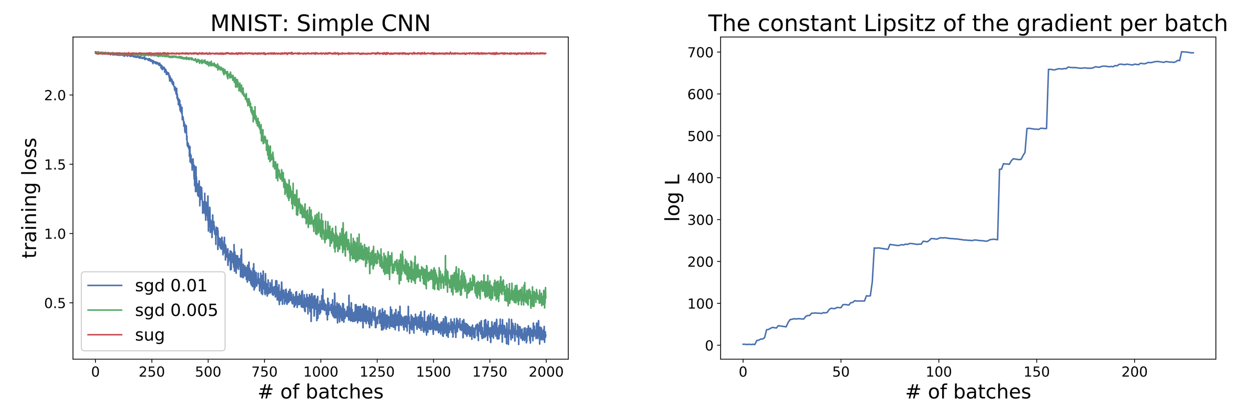
Linear Regression:



Two-layer fully-connected NN:



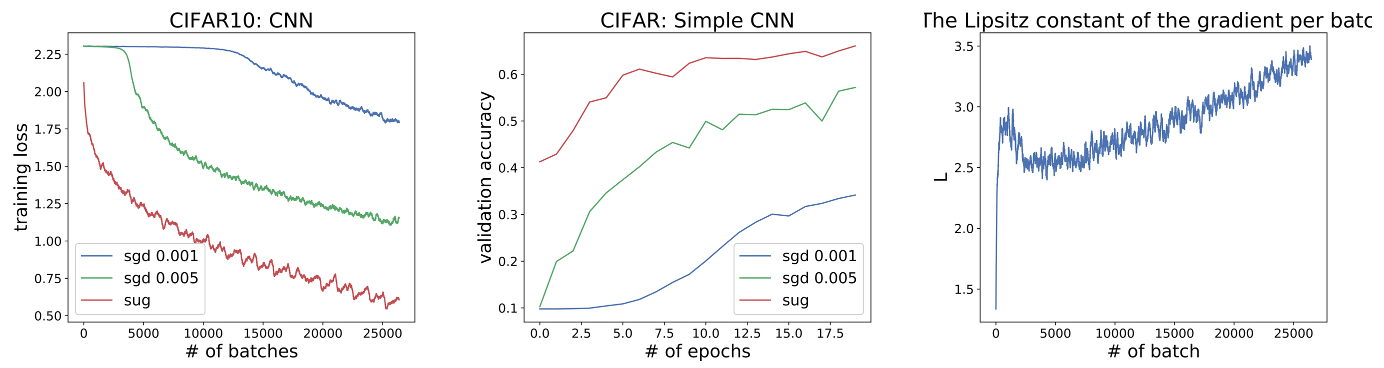
Simple convolutional NN:



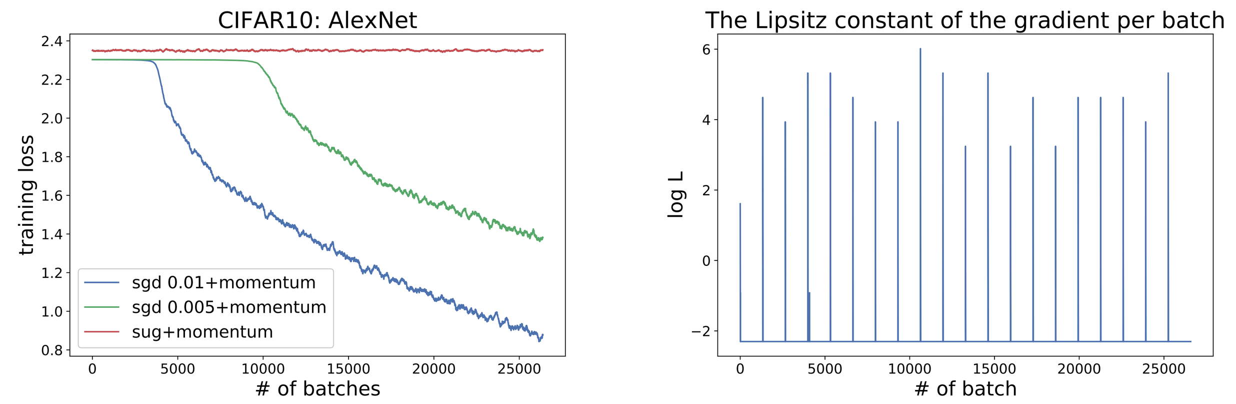
CIFAR10

The models selected for this experiment are: simple CNN, AlexNet.

Simple CNN:

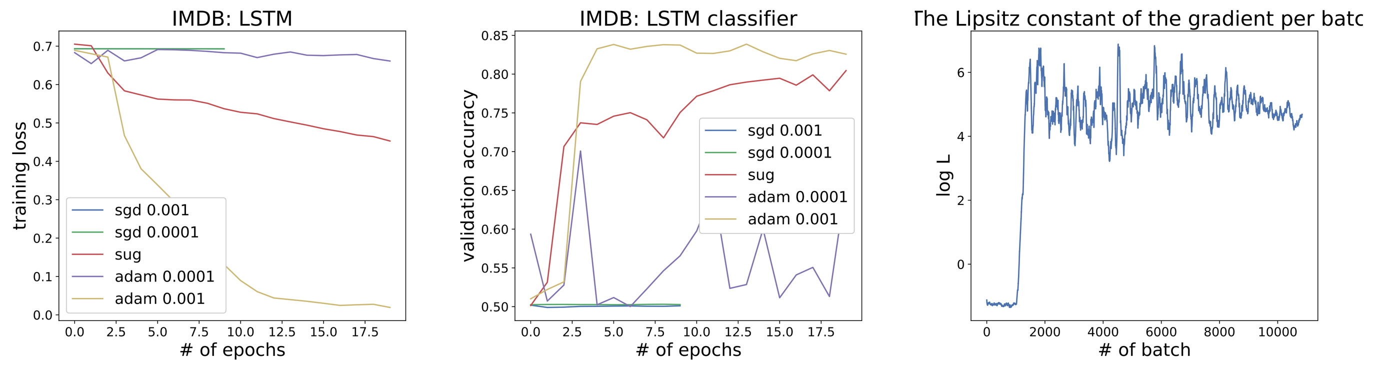


AlexNet:

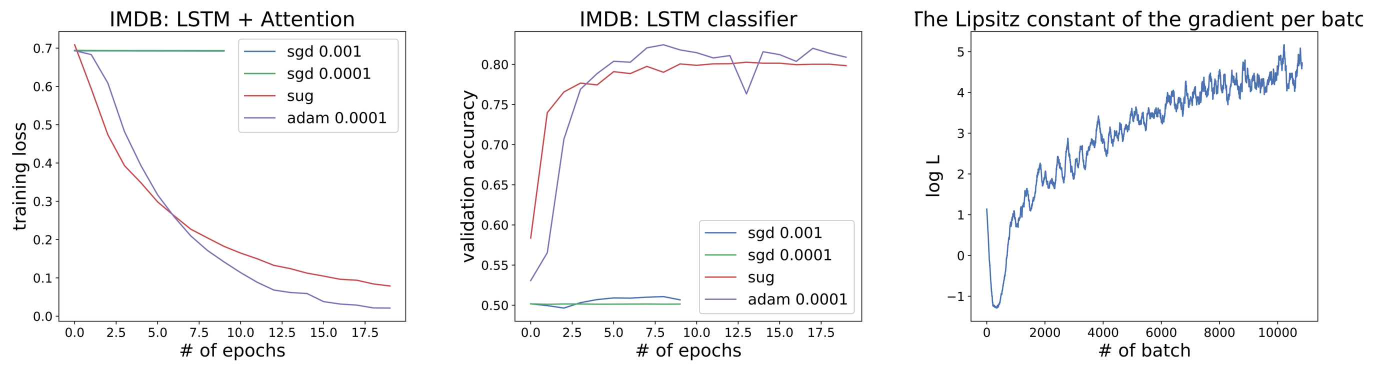


IMDb

LSTM:



LSTM with Attention:



Conclusion

The given method works well in some cases in comparison with standard SGD method, but not always.

Links

You can watch the project here