AdderNet: Do We Really Need Multiplications in Deep Learning? (Supplementary Material)

Abstract

In the main body, we propose to replace the sign gradient with full-precision gradient in AdderNets. We then analyse the convergence of taking these two kinds of gradient. Moreover, we will discuss the relationship between the ℓ_2-norm AdderNets and CNNs.

1. Convergence of Sign and Full-precision Gradient

AdderNets calculate the ℓ_1 distance between the filter and the input feature, which can be formulated as

$$Y(m,n,t) = -\sum_{i=0}^{d}\sum_{j=0}^{d}\sum_{k=0}^{c_{in}} |X(m+i,n+j,k) - F(i,j,k,t)|. \tag{1}$$

The partial derivative of Y with respect to the filters F is:

$$\frac{\partial Y(m,n,t)}{\partial F(i,j,k,t)} = \text{sgn}(X(m+i,n+j,k)-F(i,j,k,t)), \tag{2}$$

where $\text{sgn}(\cdot)$ denotes the sign function and the value of the gradient can only take +1, 0, or -1. Since Eq. (2) almost never takes the direction of steepest descent and the direction only gets worse as dimensionality grows, we propose to use the full-precision gradient:

$$\frac{\partial Y(m,n,t)}{\partial F(i,j,k,t)} = X(m+i,n+j,k) - F(i,j,k,t). \tag{3}$$

Proposition 1. *Denote an input patch as $x \in \mathbb{R}^n$ and a filter as $f \in \mathbb{R}^n$, the optimization problem is:*

$$\textit{arg}\min_{f} |x - f|. \tag{4}$$

Given a fixed learning rate α, this problem basically cannot converge to the optimal value using sign grad (Eq. (2)) via gradient descent.

Proof. The optimization problem 4 can be rewritten as:

$$\text{arg}\min_{f_1,\ldots,f_n} \sum_{i=1}^{n} |x_i - f_i|, \tag{5}$$

where $x = \{x_1, ..., x_n\}, f = \{f_1, ..., f_n\}$. The update of f_i using gradient descent is:

$$f_i^{j+1} = f_i^j - \alpha\text{sgn}(f_i^j - x_i), \tag{6}$$

where f_i^j denotes the f_i in jth iteration. Without loss of generality, we assume that $f_i^0 < x_i$. So we have:

$$f_i^{j+1} = f_i^j + \alpha = f_i^{j-1} + 2\alpha = ... = f_i^0 + (j+1)\alpha, \tag{7}$$

when $f_i^j < x_i$. Denote $t = \arg\max_j f_i^j < x_i$, we have $f_i^{t+1} >= x_i$. If $f_i^{t+1} = f_i^0 + (t+1)\alpha = x_i$ (*i.e.* $\frac{(x_i - f_i^0)}{\alpha} = t+1$), $|f_i - x_i|$ can converge to the optimal value 0. However, if $f_i^{t+1} > x_i$, we have

$$f_i^{t+2} = f_i^{t+1} - \alpha\text{sgn}(f_i^{t+1} - x_i) = f_i^0 + (t+1)\alpha - \alpha = f_i^t \tag{8}$$

Similarly, we have $f_i^{t+3} = f_i^{t+1}$. Therefore, the inequality holds:

$$f_i^{t+2k} = f_i^t < x_i < f_i^{t+2k+1}, k \in \mathbb{N}^+ \tag{9}$$

which demonstrate that the f_i cannot converge and have an error of $x_i - f_i^t$ or $x_i - f_i^t$. The f_i^j can converge to x_i if and only if $\frac{(x_i - f_i^0)}{\alpha} \in \mathbb{Z}$, which is a strict constraint since $x_i, f_i, \alpha \in \mathbb{R}$. Moreover, the f can converge to x if and only if $\frac{(x_i - f_i^0)}{\alpha} \in \mathbb{Z}$ for each $f_i \in f$. The difficulty of converge increases when the number n grows. In neural networks, the dimension of filters is can be very large. Therefore, problem 4 basically cannot converge to its optimal value. □

The aim of filters is to find the most relevant part of input features, which meets the goal of Eq. (4). The α (*i.e.* the learning rate of neural networks) can be seen as fixed when using multi-step learning rate, which is widely used in the training. According to the Proposition 1, if we use the sign gradient, the AdderNets will achieve a poor performance.

Proposition 2. *For the optimization peoblem 4, f can converge to the optimal value using full-precision gradient (Eq. (3)) with a fixed learning rate α via gradient descent when $\alpha < 1$.*

Proof. The optimization problem 4 can be rewritten as:

$$\arg\min_{f_1,...,f_n} \sum_{i=1}^{n} |x_i - f_i|, \tag{10}$$

where $x = \{x_1, ..., x_n\}, f = \{f_1, ..., f_n\}$. The update of f_i using gradient descent is:

$$f_i^{j+1} = f_i^j - \alpha(f_i^j - x_i), \tag{11}$$

where f_i^j denotes the f_i in jth iteration. If $f_i^j < x_i$, then we have the inequality:

$$f_i^{j+1} = f_i^j - \alpha(f_i^j - x_i) = (1-\alpha)f_i^j + \alpha x_i < x_i, \tag{12}$$

and $f_i^{j+1} < f_i^j$. Without loss of generality, we assume that $f_i^0 < x_i$. Then f_i^j is monotone and bounded with respect to j, so the limit of f_i^j exists and $\lim_{j\to+\infty} f_i^j \leq x_i$. Assume that $\lim_{j\to+\infty} f_i^j = l < x_i$. For $\epsilon = \alpha(x_i - l)$, there exists k subject to $l - f_i^k < \epsilon$. Then we have:

$$\begin{aligned} f_i^{k+1} &= f_i^k + \alpha(x_i - f_i^k) \geq f_i^k + \alpha(x_i - l) \\ &> l - \epsilon + alpha(x_i - l) = l, \end{aligned} \tag{13}$$

which is a contradiction. Therefore, $\lim_{j\to+\infty} f_i^j \geq x_i$. Finally, we have $\lim_{j\to+\infty} f_i^j = x_i$, *i.e.* f can converge to the optimal value. □

Therefore, by utilizing the full-precision gradient, the filters can be updated precisely.

2. Relationship Between ℓ_2-norm and Cross-correlation

In the main body, we propose to use a partial derivative in AdderNets, which is a clipped version of ℓ_2-distance. Therefore, we further discuss using the ℓ_2-distance in AdderNets instead of ℓ_1-distance. By calculating ℓ_2 distance between the filter and the input feature, the filters in ℓ_2-AdderNets can be reformulated as

$$Y(m,n,t) = -\sum_{i=0}^{d}\sum_{j=0}^{d}\sum_{k=0}^{c_{in}} \left[X(m+i,n+j,k) - F(i,j,k,t)\right]^2. \tag{14}$$

We also use the adaptive learning rate for the ℓ_2-AdderNets, since the magnitude of the gradient w.r.t X in ℓ_2-AdderNets would also be small. Table 1 shows the classification results on the ImageNet dataset. The ℓ_2-AdderNet can achieve almost the same accuracy with CNN. In fact, the output of the ℓ_2-AdderNets can be calculated as

$$\begin{aligned} Y_{\ell_2}(m,n,t) &= -\sum_{i=0}^{d}\sum_{j=0}^{d}\sum_{k=0}^{c_{in}} \left[X(m+i,n+j,k) - F(i,j,k,t)\right]^2 \\ &= \sum_{i=0}^{d}\sum_{j=0}^{d}\sum_{k=0}^{c_{in}} [2X(m+i,n+j,k) \times F(i,j,k,t) \\ &\quad - X(m+i,n+j,k)^2 - F(i,j,k,t)^2] \\ &= 2Y_{CNN}(m,n,t) - \sum_{i=0}^{d}\sum_{j=0}^{d}\sum_{k=0}^{c_{in}} [X(m+i,n+j,k)^2 \\ &\quad + F(i,j,k,t)^2]. \end{aligned} \tag{15}$$

$\sum_{i=0}^{d}\sum_{j=0}^{d}\sum_{k=0}^{c_{in}} F(i,j,k,t)^2$ is same for each channel (*i.e.* each fixed t). $\sum_{i=0}^{d}\sum_{j=0}^{d}\sum_{k=0}^{c_{in}} X(m+i,n+j,k)^2$ is the ℓ_2-norm of each input patch. If this term is same for each patch, the output of ℓ_2-AdderNet can be seen as a linear transformation of the output of CNN. Although this assumption may not always be valid, the result in Table 1 that the performance of ℓ_2-AdderNet and CNN are similar indicates that ℓ_2-distance and cross-correlation have same ability to extract the information from the inputs.

Table 1. Classification results on the ImageNet dataset using ResNet-18 model.

Method	#Mul.	#Add.	Top-1 Acc.	Top-5 Acc.
ℓ_2-AddNN	1.8G	3.6G	69.6%	89.0%
ℓ_1-AddNN	0	3.6G	66.8%	87.4%
CNN	1.8G	1.8G	69.8%	89.1%