International Core Journal of Engineering Volume 6 Issue 11, 2020
ISSN: 2414-1895 DOI: 10.6919/ICJE.202011_6(11).0038

Sketching Based Convolution Neural Network

Chen Liao[1], Mingjie Chen[2], Yeqing Wang[3], Junxiang Shi[4]

[1]School of computer science with artificial intelligence, University of Nottingham Ningbo China, Ningbo, 315100, China;

[2]South Kent school, Kent, 06785, United States;

[3]Faculty of Brain Sciences, University College London, London, WC1E 6BT, United Kingdom;

[4]West Vancouver Secondary School, West Vancouver, V7W 3H7, Canada.

Abstract

Convolutional neuron network, as an intuitive and effective approach, prevails in various industries such as computer vision and graphical information processing. However, due to the intrinsic computational complexity of convolutional operations and dataset structure (coming from the magnitude and the resolution of images and video streams etc.), CNNs could become resource consuming and relatively low in efficiency. In this paper, a novel approach will be proposed to strive a decrement on computational workload of general convolutional neuron networks based on the idea of tensor sketch. This approach is designed to replace traditional pooling layers in CNNs with sketch layers and involves unrolling convolutional layers as a fast alternation. Experiments regarding to running time and accuracy on classic image classification tasks are conducted to verify the validity of it. Though decrease in classification accuracy is detected, a significant increase in efficiency proves the prospect of this approach.

Keywords

Convolutional neuron network; Tensor sketch; Unrolling convolution; Image classification; Dimensionality reduction.

1. Introduction

Data-rich fields of artificial intelligence, such as image recognition, speech recognition and computer vision heavily emphasize on data processing, especially dealing with different types and modes of data. Deep neural networks are thus introduced which achieve promising performance when processing big data. Among them, convolutional neural networks (CNNs) have been essential to the manipulations of image data as well as steam data. As in practical situations, the power of these networks can be clearly seen in fields such as time series prediction and signal identification for unidimensional CNN as well as image classification and object detection for two-dimensional CNN [1]. Additionally, the application of CNNs could also be expanded to higher dimensions, especially for video data with time series [1]. However, as the power of CNNs become more invasive, the amount of data that goes into each network also increases dramatically. With current CNNs processing millions and billions of data, it takes up a lot of running time and storage space to process such intensive calculations. This issue naturally brings the industry into the question of model compression of CNNs. There are multiple possible strategies to simplify, or to compress a model into one that smaller in scale and low-rank approximation is a representative one. The idea of low-rank approximation considers the weight matrix of the original network as the full-rank matrix and then decomposes it into a low-rank matrix to approximate the effect of the full-rank matrix [1]. In this way intensive calculations could be simplified to a great extent using this network approximation approach

ISSN: 2414-1895 DOI: 10.6919/ICJE.202011_6(11).0038

and the running time, storage space and GPU capacity be expected decreasing [2]. Therefore, with the attempt of utilizing tensor sketch as a strategy of simplification [3], a novel model of network approximation is proposed.

In the work, the model is implemented by two main building blocks, one is the sketch-pooling layer which replaces traditional convolutional pooling approach with tensor-sketch based approach, the other is the involvement of unrolling convolutional layer, also known as fast convolutional layer. Experiments of this model are subsequently conducted with respect to aspects of running time and accuracy on the very classic LeNet-5 network for handwritten digit recognition task. Additionally, various sketching methods are also involved and compared in this experiment such as Count Sketch and Gaussian Sketch [4,5]. Finally, the experimental result is evaluated and the future prospect of this orientation is discussed.

2. Related work

There are of course many methods of network approximation and dimensionality reduction techniques, such as single value decompositions (SVD) and universal network approximation theorems. Many objectives of these techniques apply to CNNs as well, but in this paper, tensor sketch techniques will be implemented to CNNs to different real data networks to evaluate the effectiveness of deep neural network approximation using sketching techniques. Many sketching techniques could be implemented in CNNs for network approximation as well, however, they usually focus on vectors and matrices, which are less applicable for CNNs. Since sketching techniques could be extended to higher-order tensors, as previously done in works of [6], the possibility of network sketching is explored. In this paper, our works and experiments with different sketching techniques including count sketch, gaussian sketch are discussed.

Sketching is a randomized dimensionality-reduction method for data sets that aims to keep hold of similar and relevant features in the original data set. A method of sketching, count-sketch utilizes randomized hash functions to achieve such a goal by multiplying the current data set to one with one degree less. It is a probabilistic data structure that generates a table featuring the frequencies or counts of a dataset and maps them out into data structures. A Count Sketch data structure uses hash functions to map out events to frequencies, and the actual data structure itself is a two-dimensional array of K * N matrix whereas K is the number of Hash functions, and there are N columns. Each Hash function corresponds to a single row. Therefore, matrix S is created with the insurance of sparsity. A Gaussian Sketch data structure uses a similar approach to Count Sketch, and it is simply a product of a Count Sketch Matrix and a Gaussian Matrix [7].

3. Preliminary

3.1 Convolution

Convolutions are primary operations on input features and kernels in convolutional layers and pooling layers. There are three kinds of convolution operation known as ‘full’, ‘same’ and ‘valid’ which vary in size of the convolution result.

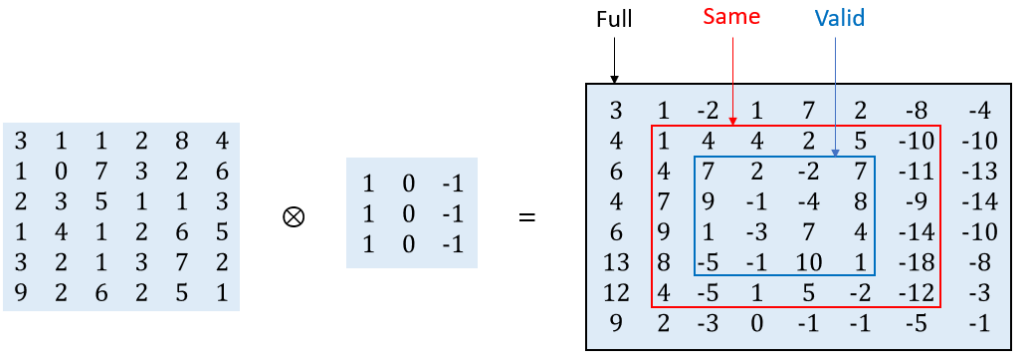

Figure 1: three types of convolutions

ISSN: 2414-1895 DOI: 10.6919/ICJE.202011_6(11).0038

The 'full' convolution enables the kernel to stretch out of the valid region of the feature map by zero padding, generating results with 'full' size. The 'same' convolution reserves the central part of a full convolution which conform to the size of the larger input. The valid convolution forbids kernels from stretching out of the valid feature region, which delivers outputs with 'valid' size. In our approach, although inputs are padded with zeros during the data preprocessing, valid convolutions are adopted in all pure convolutional operations in the forward propagation phase.

As the principle component of CNNs, convolutional operation serves as a pivotal step for feature extraction by applying filter matrices with various numerical properties [1]. Because of these properties, filters could complete tasks such as boundary detection and noise elimination. The outputs of this kind of convolution process are named feature map. There is a special kind of convolution appearing frequently in CNNs called pooling. A pooling layer serves as compacting features to decrease computational complexity within in a network. Traditional pooling strategies includes average pooling and max pooling, whereas max pooling becomes dominant recently since it underlines most significant features within a small feature neighborhood instead of blurs them (which an average pooling always does). However, the essence of a pooling operation is a convolution with a greater stride (usually greater than one), which also contributes to time redundancy caused by convolutions. Therefore, the idea of tensor sketch is naturally involved in this work to replace the whole pooling layer for a further improvement in efficiency.

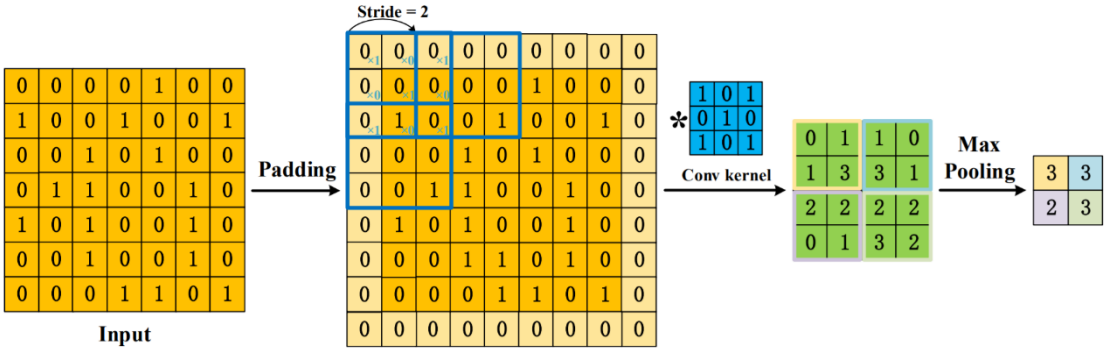

Figure 2: general convolutional process within CNN [1]

3.2 Unrolling convolution

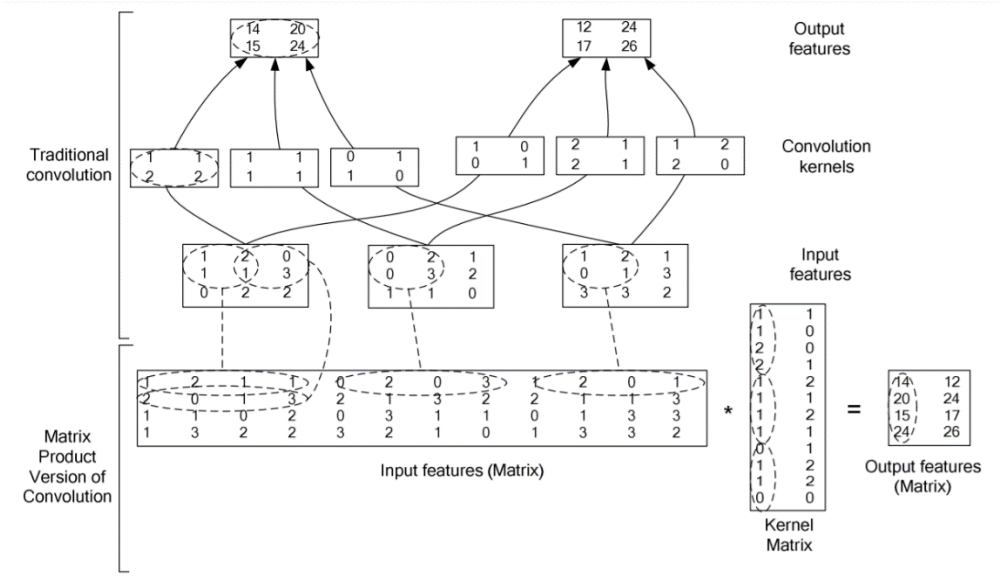

Figure 3: unrolling convolution [8]

ISSN: 2414-1895 DOI: 10.6919/ICJE.202011_6(11).0038

Since trivial convolution operation delivers an asymptotic complexity of $O(n^4)$ whereas matrix multiplication finishes in $O(n^3)$ in the worst case, it is naturally considered conducive to running time efficiency by replacing plain convolution operations with matrix multiplications. Here the idea of unrolling convolutional layers in introduced. According to Chellapilla et al, the central idea of unrolling convolution is an unfolding and duplication of the input and a rearrangement of the kernel parameters that produces a CPU friendly ordering [8]. To depict a traditional convolution to matrix multiplication, all possible unique positions of the kernel on the feature map should be identified as sub matrices. Then these matrices are reshaped into row-wise vectors. Finally, these vectors are stacked together in column-wise fashion. Similar operations are conducted on kernels but in a transposed fashion. A more intuitive illustration of transmitting process is shown in figure 3 by taking sliced 2-D matrices as an example. This set of processes converts original matrices into corresponding convolutional matrices whose multiplications are equivalent to convolutions. A more efficient convolutional layer could be acquired by applying this strategy during forward propagation [8].

3.3 Tensor sketch

As mentioned, tensor sketch is basically a dimensionality reduction approach by tensor approximation, which represents the original tensor by a smaller one within some tolerable error. This approach compacts essential information by projecting it into proper subspace [9], thus serves for the same function as traditional pooling theoretically. Therefore, max pooling layer is replaced with multiple tensor sketch strategies in our approach. With the definition from David P. Woodruff [9] and Shiva P. Kasiviswanathan [3], tensor sketch could be formally represented as:

Definition (Mode-n Sketch). *Given a tensor,* $\mathcal{T} \in \bigotimes_{i=1}^{P} \mathbb{R}^{d_i}$*, the mode-n sketch with respect to a random scaled sign matrix* $U_n \in \mathbb{R}^{k \times d_n}$ *for* $n \in [p]$*, is defined as the tensor* $S_n = \mathcal{T} \times_n U_n$ [3].

A detailed sketching process for an arbitrary tensor along a certain dimension could be calculated with the multiplication results of the sketching tensor with every possible fiber along this dimension in the tensor, where tensors along a certain dimension could be defined as vectors elicited by fixing other dimensions with values of arbitrary random combinations. A more intuitive demonstration of diverse orienting fibers of a 3-D tensor is shown in Figure 4 and how it affects the sketching process is shown in figure 5.

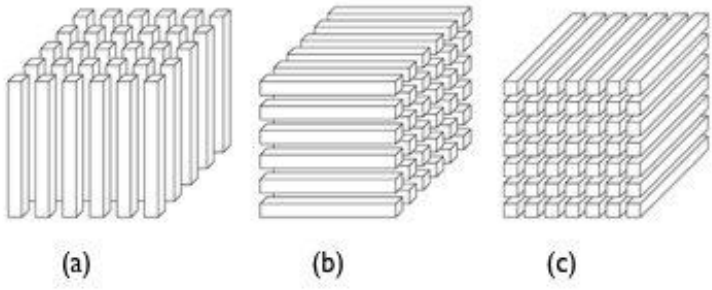

Figure 4: fibers along each dimension [10]

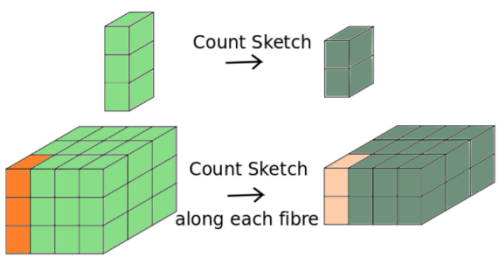

Figure 5: Example of count sketch along each fiber [10].

ISSN: 2414-1895 DOI: 10.6919/ICJE.202011_6(11).0038

Various sketching strategies are available and proved effective, but here only Gaussian Sketch (denoted as GS for simplicity) and Count Sketch (denoted as CS for simplicity) are scrutinized and experimented. A very basic difference of CS and GS lies in the sketching matrix. Within the same size as a premise, the sketch matrix of CS consists of columns with one and only one random binary signal of value either 1 or -1 in a random row where as the one of GS is a matrix of normal random variables with an average of zero and a certain standard deviation. Typical representations of sketch matrices of both CS and GS are shown in Figure 6. Comparing to sketch matrix of GS, sketch matrix of CS is priory in its sparsity, which could allow sparse matrix calculation for even a better efficiency than GS. Sparse matrix manipulation is already a mature technique and well supported by various libraries such as Scipy and Matlab sparse library.

$$\begin{bmatrix} 0.54 & 0.32 & 3.58 & 0.73 & -0.12 & 0.67 & 0.49 & 0.29 \\ 1.83 & -1.31 & 2.77 & -0.06 & 1.49 & -1.21 & 1.03 & -0.79 \\ -2.26 & -0.43 & -1.35 & 0.71 & 1.41 & 0.72 & 0.73 & 0.89 \\ 0.86 & 0.34 & 3.03 & -0.20 & 1.42 & 1.63 & -0.3 & -1.15 \end{bmatrix} \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 1 & 0 & -1 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Figure 6: gaussian sketch matrix (left) and count sketch matrix (right)

4. Proposed Model

The differences of this model comparing to traditional ones comes from two aspects: the replacement of plain convolution operation with matrix multiplication and the replacement of pooling layers with sketching layers. In the unrolling convolutional layer, multidimensional input features and kernels are sliced firstly. Then every input feature and kernel pair is converted into a convolutional matrix pair. Subsequently, multiplications are applied and results are stacked. Finally, the result tensor is reorganized to conform to the result of plain convolution thus the spatial property of the original input still holds. The sequence diagram (Figure 7) shows the working flow of our layer comparing to traditional layers.

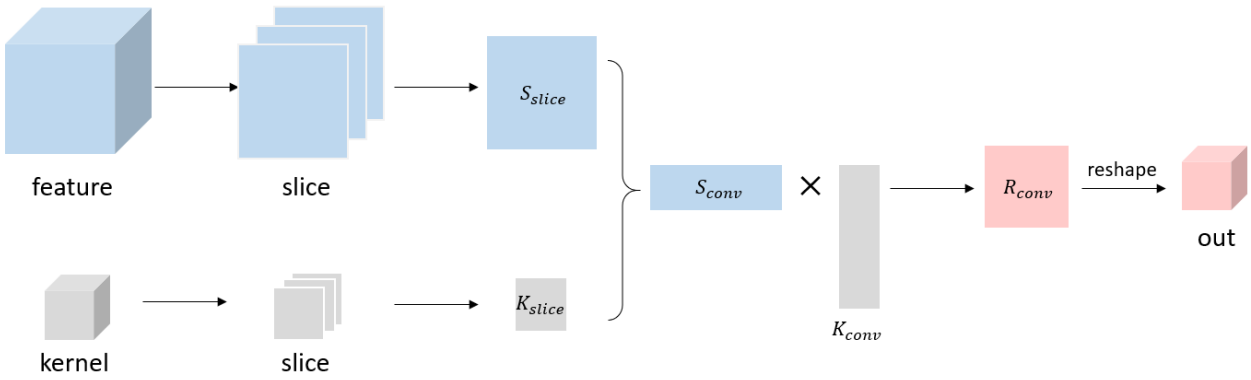

Figure 7: sequence diagram of fast convolutional layer

In the second refinement, traditional pooling layers in CNN are replaced with tensor sketches as both compress the input and make approximations but tensor sketch is prior in asymptotic complexity. Another technique applied in sketching layers is multi-way sketch which derives from the idea of higher-order count sketch [10, 11]. Since single sketch operation only compresses information along single dimension, the ratio of the input is modified and some of the spatial information is lost if we only apply single sketch on the input tensor. Therefore, multiple sketches with the same sketch matrix along diverse directions are applied to maintain the original ratio of the input. The sequence diagram (Figure 8) shows the working flow of our sketching layer comparing to traditional pooling layers. Generally, the scheme of our model follows an add-on and replicable fashion as refinements of this

ISSN: 2414-1895 DOI: 10.6919/ICJE.202011_6(11).0038

model could be transplanted to all convolutional neuron networks because modifications lie in convolutional layers and pooling layers — the most essential modules in CNNs.

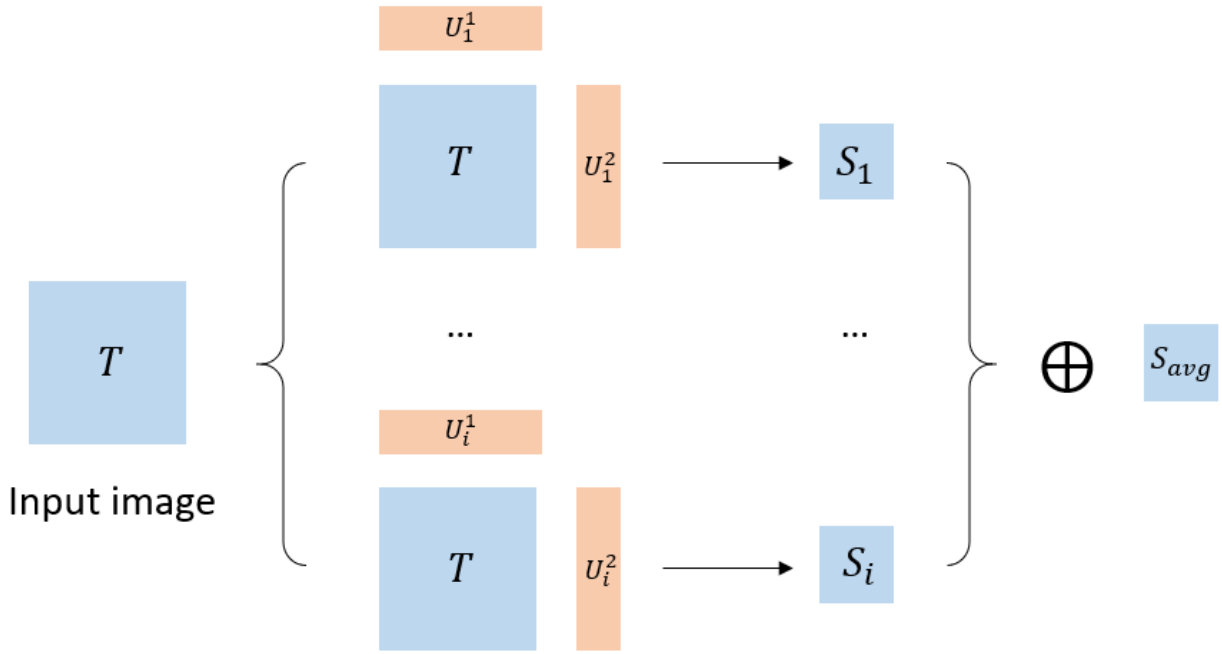

Figure 8: sequence diagram of fast pooling layer

5. Method

The proposed model is transplanted into classic LeNet model for classification tasks on MINST database aiming at the multi-class classification of 28×28 greyscale images into one of the ten-digit classes: 0-9 [12]. The network consists of 7 layers with trainable parameters respectively. Layer C1, C3 and C5 are convolution layers with different feature maps. S2 and S4 are subsampling layer corresponding to previous layer, which results are coming through a sigmoidal function. Next layer F6 is fully connected to C5. The last layer is output layer.

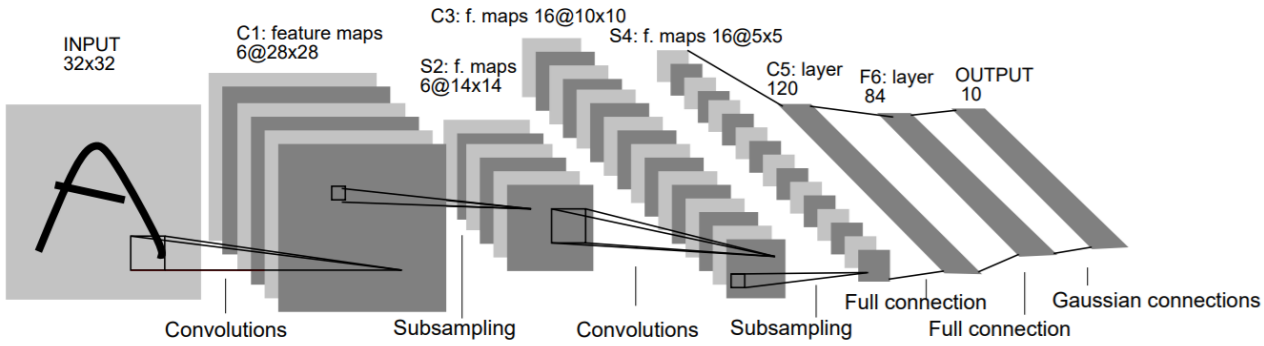

Figure 9: LeNet-5 structure [12]

6. Experiment

In this section, the effectiveness of our proposed method will be experimentally demonstrated. By applying different sketching methods on the pooling layers of the CNN Models and involving unrolling convolution, a computational-optimized network is implemented. The goal through experiments is to compare the running time and the accuracy of optimized networks and plain networks. There are three comparing groups in total, two methods were mentioned in the Background

ISSN: 2414-1895 DOI: 10.6919/ICJE.202011_6(11).0038

session, the count sketch and gaussian sketch, and one last comparing group is the plane CNN model. By doing this, the differences in running time and accuracy can be compared.

6.1 Results

Table 1. Accuracy

Batch(ith)	Fast Convolution Layer	Gaussian sketch	Count Sketch
10	21%	13%	7%
20	31%	13%	7%
30	37%	13%	12%
40	37%	13%	17%
50	49%	13%	7%
60	48%	13%	7%
70	44%	13%	7%
80	58%	13%	10%
90	57%	13%	9%
100	60%	13%	7%

Table 2. Accumulative Time Complexity

Batch(ith)	Fast Convolution Layer	Gaussian sketch	Count Sketch
10	10.4	5.97	6.36
20	20.8	12.05	12.81
30	31.39	17.95	19.19
40	41.83	24.03	25.74
50	52.06	30.23	32.09
60	63.46	36.94	38.47
70	75.06	43.4	44.94
80	86.88	49.83	51.32
90	98.95	56.24	58.04
100	110.84	62.86	64.35

The results are shown in the graph above; the model was being run ten times to compare the total final running time. Table 1 shows the accuracy result when we compare the three methods; Table 2 shows the result of total running time. Figure 10 shows the fast convolution operation running time over 100 seconds. The convolutional layer, the running time, dramatically reduces to 65 seconds. The Gaussian sketch, the running reduces to around 63 seconds. By applying proposed method, each run takes less time as well for our two intended purposes. Aside from the running time, accuracy is another essential part to consider. As Table 1 shows, the method did not achieve the same accuracy compared to the Fast Convolution itself. If the model is being trained enough, the accuracy will rise to a better level; however, the proposed method will not achieve an acceptable performance.

ISSN: 2414-1895 DOI: 10.6919/ICJE.202011_6(11).0038

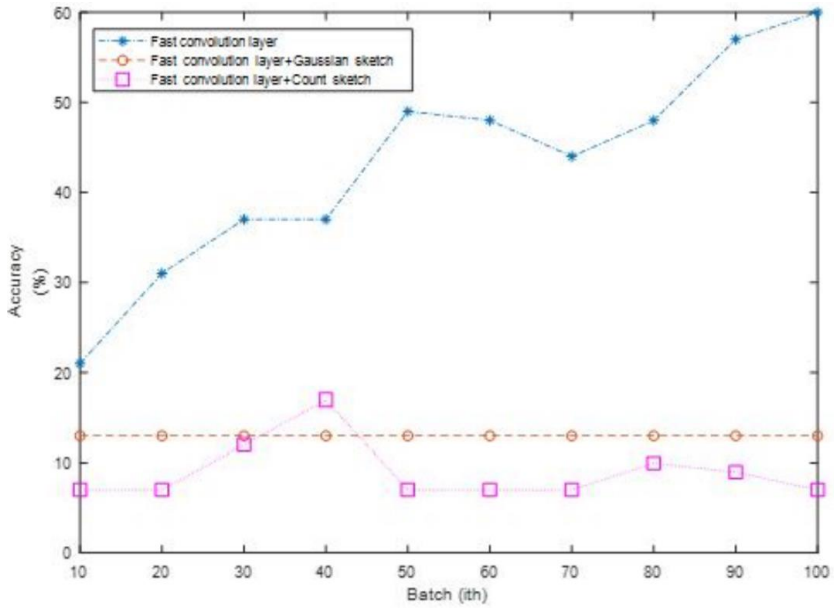

Figure 10: result visualization

7. Discussion

From the result, comparing to the group with Gaussian sketch, the running time of group Fast convolution layer with Count sketch increases in terms of running time. However, the time difference between group Fast convolution layer with Count sketch and is not so significant. As aspect of accuracy, the pure fast convolution layer is much higher than the other two groups during all training batch, which indicates inefficiency of new fusion of tensor sketch method into LeNet model.

For the future orientation of this work, another intriguing algorithm—OSNAP and random samples could also be put into comparison with Sketch methods. Additionally, our algorithm could be adjusted to adapt to other model structure on certain datasets, such as Fashion-MNIST, CIFAR-10, CIFAR-100. Furthermore, applying the innovative tensor sketch to different convolutional neural network model, for example, VGG-16 could deliver a more comprehensive understanding that whether tensor sketch method can be applied and successfully improved in CNN could be achieved.

8. Conclusion

In this paper, a novel approach of sketching-based convolution network model is proposed with the idea of sketching pooling layer and unrolling convolutional layer. As a general model, this approach is compatible with various kinds of convolutional neuron networks and could be easily transplanted. Result from various experiments indicates that though efficiency increases considerably, there remain problems on accuracy issues. The future work that entails this orientation could be the improvement on model accuracy as well as the transplant of this model to other network structure.

References

[1] Zewen Li, Wenjie Yang, Shouheng Peng, and Fan Liu. A survey of convolutional neural networks: Analysis, applications, and prospects, 2020.

[2] Jelani Nelson and Huy L. Nguyen. OSNAP: Faster numerical linear algebra algorithms via sparser subspace embeddings. arXiv e-prints, page arXiv:1211.1002, November 2012.

[3] Shiva Prasad Kasiviswanathan, Nina Narodytska, and Hongxia Jin. Deep neural network approximation using tensor sketching, 2017.

[4] David P. Woodruff, "Sketching as a Tool for Numerical Linear Algebra," in Sketching as a Tool for Numerical Linear Algebra, now, 2014.

[5] Pham, Ninh & Pagh, Rasmus. (2013). Fast and scalable polynomial kernels via explicit feature maps. 239-247. 10.1145/2487575.2487591.

ISSN: 2414-1895 DOI: 10.6919/ICJE.202011_6(11).0038

[6] Weicai Wang, Yang Gao, Pablo Iribarren, Yanbin Lei, Yang Xiang, Guoqing Zhang, Li Shenghai, and Anxin Lu. Wang et al. 2015. 09 2015.

[7] Douglas J. Orr, Andre Alc ´ antara, Maxim V. Kapralov, P. John Andralojc, Elizabete ˆ Carmo-Silva, and Martin A.J. Parry. Surveying rubisco diversity and temperature response to improve crop photosynthetic efficiency. Plant Physiology, 172(2):707– 717, 2016.

[8] Kumar Chellapilla, Sidd Puri, and Patrice Simard. High Performance Convolutional Neural Networks for Document Processing. In Guy Lorette, editor, Tenth International Workshop on Frontiers in Handwriting Recognition, La Baule (France), October 2006. Universite de Rennes 1, Suvisoft. http://www.suvisoft.com.

[9] David P. Woodruff. Computational advertising: Techniques for targeting relevant ads. Foundations and Trends R in Theoretical Computer Science, 10(1-2):1–157, 2014.

[10] Tekin, Bugra. (2013). Learning Separable Filters with Shared Parts. 10.13140/RG.2.2.30424.90888.

[11] Y. Shi and A. Anandkumar, "Higher-Order Count Sketch: Dimensionality Reduction that Retains Efficient Tensor Operations," 2020 Data Compression Conference (DCC), Snowbird, UT, USA, 2020, pp. 394-394, doi: 10.1109/DCC47342.2020.00045.

[12] Y. Lecun, L. Bottou, Y. Bengio and P. Haffner, "Gradient-based learning applied to document recognition," in Proceedings of the IEEE, vol. 86, no. 11, pp. 2278-2324, Nov. 1998, doi: 10.1109/5. 726791.