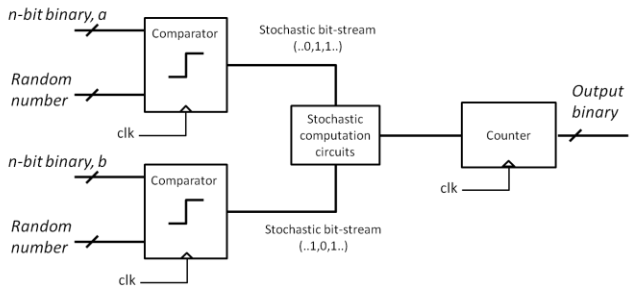
Revisiting Stochastic Computation: Approximate Estimation of Machine Learning Kernels

Suyog Gupta[†] and Kailash Gopalakrishnan[‡]
IBM T. J. Watson Research Center, Yorktown Heights, NY 10598, USA
{[†] suyog, [‡] kailash}@us.ibm.com

In this position paper, we present an argument in the favor of employing stochastic computation for approximate calculation of kernels commonly used in machine learning algorithms. The low cost and complexity of hardware implementation of stochastic computational blocks, coupled with the inherent error resilience of a wide range of machine learning algorithms, offers new and interesting avenues for realizing hardware solutions that can exploit the energy-performance-accuracy tradeoff for data analytics workloads.

The foundations of stochastic computation (SC) can be traced to the work by Poppelbaum [1] and Gaines [2] in the late 1960's. Within the stochastic computation framework, a signal value x is represented as a Bernoulli bit-sequence $X=(X_1, X_2, ..., X_m)$ such that $P(X_i = 1) = x$. For instance, $x = 5/8$ can be represented by a sequence of bits: (0,1,0,1,1,1,0,1). This representation of numbers as probabilities allows for implementation of arithmetic operations such as addition and multiplication using simple digital logic gates [3]. A logical AND of 2 bit-streams produces a bit-stream that corresponds to the product of the numbers represented by the input bit-streams. Compared with analog [eg. 4] or mixed analog-digital [eg. 5] circuits for approximate computation, SC has the advantage of seamless compatibility with the state-of-the-art digital CMOS platform. Furthermore, the accuracy of a stochastic calculation can be tuned by varying the length of the bit-stream used in the stochastic representation; the computation accuracy increases progressively with the increase in the bit-stream length. Interestingly, this control over the accuracy can be achieved without modifying the underlying hardware. As opposed to the positional number system traditionally used in binary number representation, each bit in the stochastic bit-stream is assigned the same weight. As a result, the stochastic representation is expected to be considerably more tolerant to errors arising out of random bit flips.

Despite its unique advantages, several practical considerations limit the usefulness of SC and the idea of computation using stochastic bit-streams has failed to garner wide-spread research attention. So far, SC has been limited to niche applications such as low-density parity check (LDPC) decoding [6]. Motivated by its superior fault tolerance and the considerably small hardware footprint, some research effort has also been invested in extending SC techniques to image processing applications [7, 8].



▪ No need for specialized elements beyond conventional digital CMOS primitives ▪ Accuracy can be tuned dynamically by adjusting the sequence length ▪ Very small hardware footprint ▪ Sparse data representation – *Fault tolerant: improved immunity to bit flips*	▪ Limited precision: sequence of m bits can only represent m numbers ▪ High latency: large number of clock cycles needed to generate stochastic bit-streams. ▪ Low accuracy compared with conventional computation techniques

Fig. 1. (a) A stochastic computation system. (b) Summary of the key advantages (left) and challenges (right) of stochastic computation

Historically, the primary drawback of the SC method has been the large overhead in terms of the number of clock cycles required to generate the stochastic bit-streams. A stochastic bit-stream of length m can represent numbers with a precision of $1/m$. A given stochastic representation of a number is not unique and fluctuations in the bit-stream representation contribute to the inaccuracy in SC. The error introduced as a result follows a normal distribution with

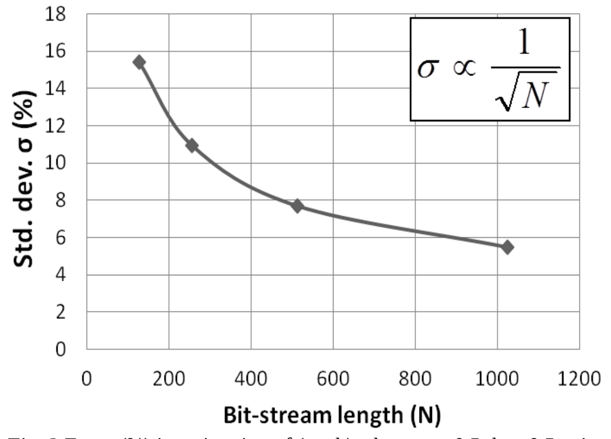


Fig. 2.Error (%) in estimation of (a x b) where a = 0.5, b = 0.5 using SC as a function of the bit stream length. Error statistics are obtained from 1000 trials.

zero mean and a variance that is inversely proportional to the bit-stream length used to represent the number as shown in Fig. 2. As a result, a serial implementation of the stochastic computation, although extremely area-efficient, suffers from high latency due to the long bit-streams required to achieve reasonable accuracy. This drawback has motivated a recent effort [9] aimed towards developing a parallel stochastic computing architecture that improves the computation speed and accuracy at the cost of increasing the area footprint of the overall system.

It is worth noting that the dimension of the input data also plays an important role in determining the accuracy of the SC result. For instance, the stochastic computation of the inner product of two d-dimensional vectors requires generating N-bit long sequences for each dimension of the two input vectors. The inner product can then be computed by counting the occurrence of '1' in the d x N output bits and normalizing the result by N. In such a case, the variance of the percentage error in the inner product calculation using SC is inversely proportional to the quantity d x N. The chart in Fig. 3 quantifies this observation. For computations on datasets with high dimensionality, it is then possible to significantly reduce the length N of the stochastic bit-stream while maintaining the same degree of accuracy.

Applications that can tolerate a certain degree of inexactness in the underlying computation are very well-suited for SC. As analyzed in the recent work [10], a wide range of machine learning algorithms designed specifically for pattern recognition, data mining and search are suitable for approximate computing. In most of these algorithms, a single computational kernel can be identified that is responsible for a large fraction of the algorithm run time. Typically, these algorithms are required to work with very high dimensional datasets. As an example, consider the MNIST handwriting recognition database [11] commonly used for benchmarking the performance of classification algorithms. Each data point in this database is a feature vector in a 784-dimensional space. The inner product of two such data points forms the dominant computational kernel in the linear support vector machine classification algorithm and may be computed approximately using SC.

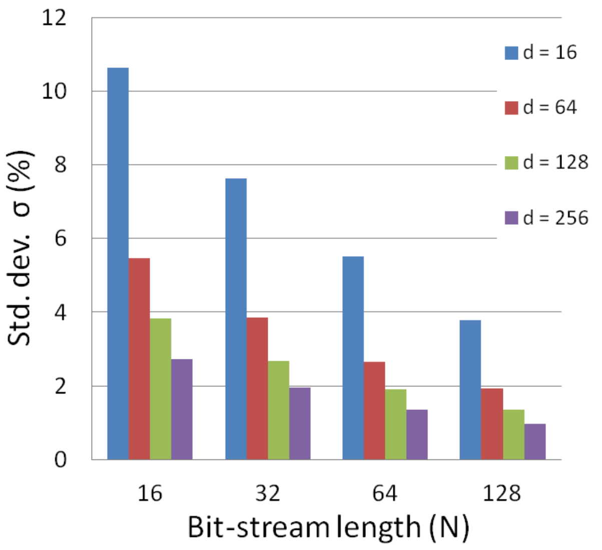


Fig.3. Error (%) in estimation of A^TB where A, B are d-dimensional vectors with each component = 0.5. Error statistics are obtained from 1000 trials.

The burgeoning research interest in the area of approximate computing provides the motivation for revisiting stochastic computation. This unconventional technique of information processing offers a new and unique set of tools for trading off computation accuracy for gain in performance and/or power consumption and warrants a more careful consideration. In a stochastic computing system, stochastic bit-stream generation represents a sizeable overhead and limits the degree of parallelism that may be achieved. Device-level and/or circuit-level innovations that reduce this overhead can improve the overall performance of a stochastic system. Furthermore, architecture-level research is needed to determine the optimal place in the system for generating stochastic bit-streams (near memory, near caches or near the core), hardware interfaces that feed data into the bit-stream generator, as well as all the components that might be needed to build a full "Stochastic ALU".

The authors would like to thank Vijayalakshmi Srinivasan and Vikas Sindhwani from IBM Research for useful discussions and insights.

References

[1] W. J. Poppelbaum, C. Afuso, J. W. Esch, "Stochastic computing elements and systems", in *Proc. AFIPS Fall Conf.*, pp. 635-644, 1967
[2] B.R. Gaines, "Stochastic Computing", *Advances in Information System Sciences,* vol. 2, no. 2, pp. 37-172, 1969.
[3] B. D. Brown and H. C. Card, "Stochastic neural computation I: Computational elements", *IEEE Trans. Comput.*, vol 50, pp. 891-905, 2001.
[4] G. Han and E. Sanchez-Sinencio, "CMOS Transconductance Multipliers: A Tutorial", *IEEE Trans. Circuits Syst.*, vol. 45, no. 12, pp. 1550- 1562, 1998.
[5] J. A. Albarran and D. A. Hodges, "A charge-transfer Multiplying Digital-to-Analog Converter", *J. Solid State Circuits,* vol. SC-11, no. 6, pp. 772-778, 1976
[6] W. J. Gross, V. C. Gaudet, and A. Milner, "Stochastic implementation of LDPC decoders", in *Proceedings of the Asilomar Conference on Signals, Systems and Computers,* pp.713–717. 2005.
[7] A. Alaghi, C. Li and J. P. Hayes, "Stochastic Circuits for Real-Time Image-Processing Application", in *Proc. Design Automation Conference,* pp. 1-6, 2013.
[8] P. Li and D. Lilja, "Using Stochastic Computing to Implement Digital Image Processing Algorithms", in *IEEE Intl. Conf. on Computer Design.* pp. 154-161, 2011.
[9] L. Miao and C. Chakrabarti, "A parallel stochastic computing system with improved accuracy", in *Proc. IEEE Workshop on Signal Processing Systems*, 2013
[10] V. K. Chippa, S. T. Chakradhar, K. Roy and A. Raghunathan, "Analysis and Characterization of Inherent Application Resilience for Approximate Computing", in *Proc. Design Automation Conference,* 2013.
[11] Y. LeCun, C. Cortes, and C. Burges, "The MNIST database of handwritten digits". http://yann.lecun.com/exdb/mnist/