Self-Organizing Infrastructure for Machine (Deep) Learning at Scale

Samir Mittal
Micron Technology
Milpitas, CA, 95035
USA
samir@micron.com

ABSTRACT

Building machine (deep) learning[1] systems is hard. Computation requirements grow non-linearly with the complexity of the task at hand creating acute challenges relating to data dimensionality, complex model development, slow experiments, and scalability of production deployments. The bulk of the ML/DL effort is consumed in infrastructure and data management. Automating such workflows has become the focus of recent research activity, so as to make ML/DL systems universally accessible. We extend these paradigms by infusing domain knowledge for infrastructure self-management. Key elements include understanding application design intent, fingerprinting the neural network for its computational, data and convergence properties, optimizing the implementation to achieving workload intent, and accelerating the neural network implementation in real-time hardware implementation. Keys to success require offline behavioural modelling coupled with online dynamic adaptation, made possible by the use of cognitive algorithms that accumulate knowledge in a dynamic and continuously evolving knowledgebase. In this way, we use machine learning to automate AI infrastructure management to minimize human engineering, and significantly accelerating application performance.

CCS CONCEPTS

• **Software and its engineering → software infrastructure**; • **Applied computing → Enterprise computing;**

[1] The term machine (deep) learning refers to generalized infrastructure requirements to support both deep learning and machine learning workloads.

SE4COG'18, May 28–29, 2018, Gothenburg, Sweden

ACM ISBN 978-1-4503-5740-1/18/05…$15.00
https://doi.org/10.1145/3195555.3195562

KEYWORDS

Deep learning, machine learning, cognitive infrastructure

ACM Reference format:

S. Mittal. 2018. In *Proceedings of the 1st International Workshop on Software Engineering for Cognitive Services (SE4COG'18) conference, Gothenburg, Sweden, May 2018*, 4 pages.
DOI: 10.1145/3195555.3195562

1 INTRODUCTION

As depicted in Figure 1, the information technology landscape has evolved from providing business intelligence through analytics to enabling operational intelligence with real-time insights. Businesses are able to generate real-time insights to harness perishable in-the-moment opportunities. These insights are synthesized through analysis of live data within the contextual lens of multi-domain historical data. Inference rules that were traditionally generated with classical ETL processes are being quickly replaced by Machine Learning (ML) and Deep Learning (DL) techniques to generate knowledge based inference engines.

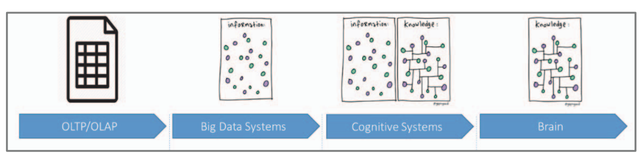

Figure 1: Evolution of Information Technology landscape (from [1]).

Advancements in knowledge based learning systems are accelerating. A rich eco-system of algorithms and open source frameworks, implemented on commodity infrastructure, holds the promise of universally accessible capability. Society scale impact is expected in autonomous driving, contextualized recommendations, personalized medicine, and other verticals. As shown in Figure 2, machine translation capability is approaching human quality, and automated large-scale image classification is beating human capability (see Figure 3). In medicine, ML/DL systems have started to outperform doctors in detecting breast cancer in radiology images [2] It is the dawn of the golden age of data, and we have only begun to unlock key capabilities through learning structures.

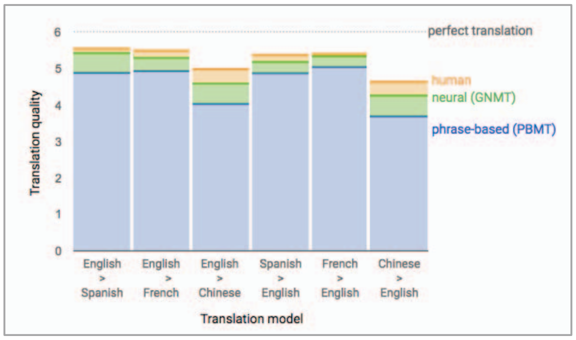

Figure 2: DL/ML language translation capability (from [3]).

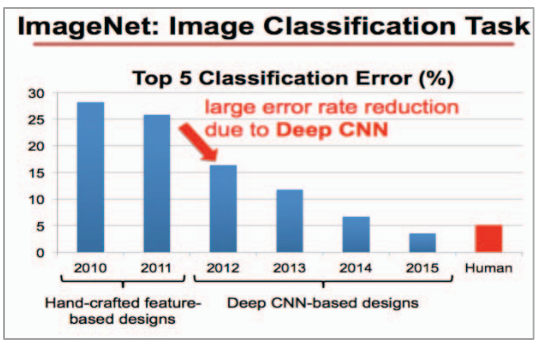

Figure 3: Automated image classification (from [4]).

2 CHALLENGES WITH MACHINE LEARNING

Building machine learning systems is hard. The most visible and the best success stories require 100's if not 1000's of engineers. The prediction accuracy of learning systems improves with more data and larger models. Computation requirements grow non-linearly with the complexity of the task at hand (Figure 4). This creates acute challenges relating to data dimensionality, complex model development, slow experiments, and scalability of production deployments

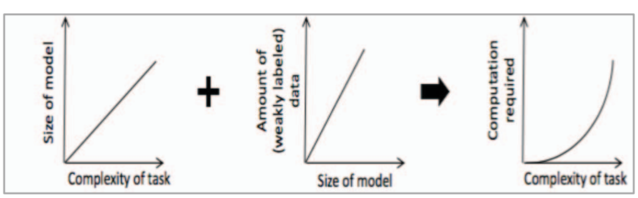

Figure 4: Computational requirements for scale-out learning (from [5]).

The data and the computation pipelines in DL/ML systems are complex. As explained in a recent paper [6], DL/ML code comprises only 10% of real world ML systems. The bulk of the effort is consumed in infrastructure and data management (Figure 5). Automating much of this pipeline has become a focus of recent research activity, so as to make ML/DL systems universally accessible, and refocus the activities of the domain expert in building production quality systems [7].

Training ML/DL systems are compute intensive tasks, where models can take exaflops to compute while processing and generating petabytes of input and intermediate data. The compute complexity is high; medium sized experiments and popular benchmarks can take days to run [8], severely compromising the productivity of the data scientist. Distributed scaling stalls only after a dozen nodes due to locking, messaging, synchronization and data locality issues. The rate of data avalanche is beating the growth rates from Moore's law, resulting in diminished economic returns at scale (see Figure 6).

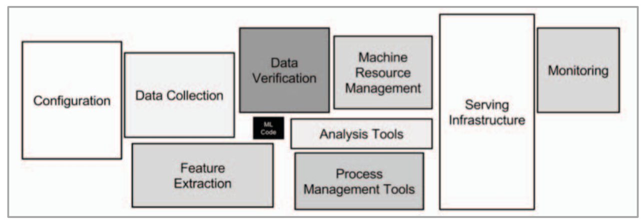

Figure 5. The hidden technical debt in machine learning systems (from [6]).

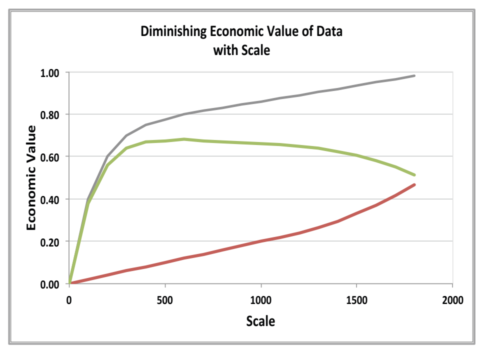

Figure 6. Diminishing economic value of data with scale.

3 SELF ORGANIZING INFRASTRUCTURE

To address the issues highlighted above, recent research from Rao and others has turned to self-managing, self-aware models of computing [9]. The idea is to enable autonomic management of the compute infrastructure to the application intent with regards to application performance and data security. Policies defining resource, workload and computation process rules regulate application performance (see Figure 7).

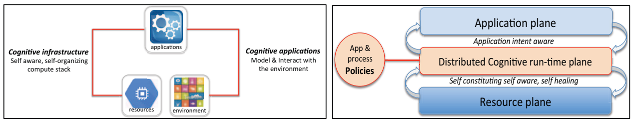

Figure 7. Cognitive distributed computing model by Dr. Rao Mikkilineni [11].

We enable this paradigm by infusing domain knowledge for self-management in a sequential and hierarchical process depicted in Figure 8. Key elements of this process include the ability to understanding application design intent, fingerprinting the network for its computational, data and convergence properties, optimizing the neural network implementation to achieve workload intent, and accelerating the network implementation in real-time hardware. Keys to the success require offline behavioral modeling coupled with online dynamic adaptation, made possible by the use of cognitive algorithms that accumulate knowledge in a dynamic and continuously evolving knowledgebase. In this sense, using machine learning to automate AI infrastructure management minimizes human engineering, with the potential to significantly accelerating application performance.

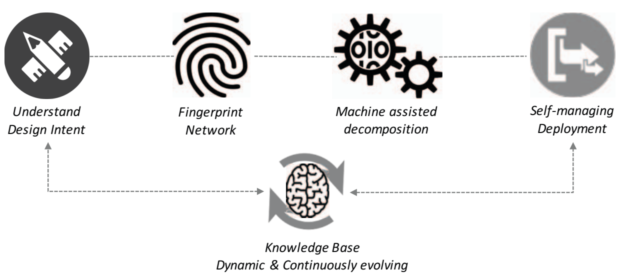

Figure 8. Infusing domain knowledge for infrastructure self-management.

This approach resolves many of the canonical issues in the today's AI stack. As depicted in Figure 9, the state of the art is human engineering intensive. Inefficiencies with respect to neural network decomposition, distributed data management and model convergence and tune-up are pushing infrastructure to its limits with weak parallelization, data mobility and locality issues and poor use of hardware knowledge. In contrast, we propose to characterize the workflows with regards to their neural network computation characteristics, data dependencies and the neural network convergence properties. Using a policy based engine, we match the workload non-functional specifications to the hardware attributes through a run-time optimizer that accelerates workflows through optimized partitioning, predictive data movement and intelligent scale-out. In this way, we propose a new generation of optimizing compilers using offline and online AI to eliminate the gap between the neural network specification and its execution, and significantly improving infrastructure performance.

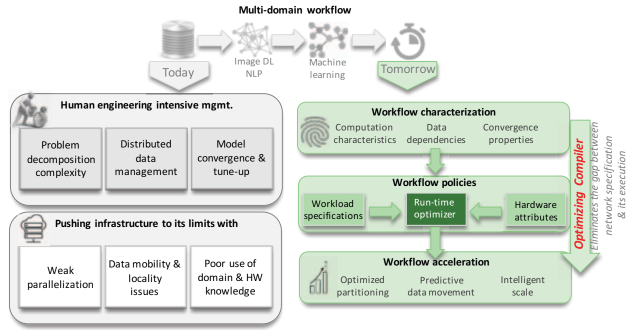

Figure 9. Infusing domain knowledge for infrastructure self-management.

4 CONCLUSIONS

Critical requirements to enable self-organizing infrastructure include automated domain modeling from large candidate data sets, automated performance optimization, and automatic adapation to accommodate fluctuations. Despite these challenges, this strategy promises exciting and new results such as those reported in [10]. It enables Separation of Concerns so that the infrastructure and process optimization concerns are separated from application logic to hide complexity that allows domain experts to re-focus on ML/DL breakthroughs.

Self-organizing infrastructure enables autonomic scaling to economize scaling, and allow applications to achieve high performance without human engineering. It extracts the best performance of many-core systems, while optimizing for reliability and data movement—the primary impediments in designing scale-out DL/ML systems.

Self-organizing infrastructure has the ability to handle complexity with scale. The system can accumulate knowledge and act on it to adaptively tune its behavior to robustly achieve desired goals within the performance, power, and resilience envelopes specified in the policy framework.

Self-organizing infrastructure should provide computational resiliency and trusted results as it improves the resiliency of data, applications, software, and hardware systems, as well as the trustworthiness of the results produced through in-situ fault detection, fault prediction and trust verification mechanisms.

REFERENCES

[1] Balint Fleischer and Jian-Li. 2016 Storage Implications of Cognitive Computing. Available online at http://www.snia.org/sites/default/DSI/2016/presentations/Balint-Fleischer-JianLi_Storage_Implications_Cognitive_Computing_1-04.pdf
[2] Y. Liu, K. Gadepalli, et al. 2017. Detecting Cancer Metastases on Gigapixel Pathology Images. *arXiv 2017.* arXiv:1703.02442
[3] Quoc Le and Mike Schuster. 2016. A Neural Network for Machine Translation at Production Scale. Available online at https://research.googleblog.com/2016/09/a-neural-network-for-machine.html
[4] O. Russakovsky, J. Deng, et al. 2015. Imagenet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision* (2015), 115, 211–252.
[5] T. M. Chilimbi, Y. Suzue, et al. Project Adam. 2014. Building an Efficient and Scalable Deep Learning Training System. In *Proceedings of the 11th Usenix Conference on Operating Systems Design and Implementation*, Broomfield, CO, USA, 6–8 October 2014; Volume 14.

[6] D. Sculley, G. Hold, et al. 2015. Hidden Technical Debt in Machine Learning Systems. In *Proceedings of the 28th International Conference on Neural Information Processing Systems*, Montreal, QC, Canada, 7–12 December 2015; 2503–2511.

[7] S. Palkar, J.J. Thomas, et al. 2017. Weld: A Common Runtime for High Performance Data Analytics. In *Proceedings of the Conference on Innovative Data Systems Research (CIDR)*, Chaminade, CA, USA, 8–11 January 2017.

[8] J. Keuper, F.-J. Preundt. 2016. Distributed Training of Deep Neural Networks: Theoretical and Practical Limits of Parallel Scalability. In *Proceedings of the Workshop on Machine Learning in High Performance Computing Environments*, Salt Lake City, UT, USA, 13–18 November 2016.

[9] R. Mikkilineni, A. Comparini, G. Morana. 2012. The Turing O-Machine and the DIME Network Architecture: Injecting the Architectural Resiliency into Distributed Computing. *In Proceedings of the Turing-100—The Alan Turing Centenary*, Manchester, UK, 22–25 June 2012.

[10] U. S. Department of Energy. 2015. Machine Learning and Understanding for Intelligent Extreme Scale Scientific Computing and Discovery, *DOE Workshop Report*, Rockville, MD, USA, 2015.

[11] Rao Mikkilineni. Available online at https://www.linkedin.com/in/raomikkilineni/