FEEDFORWARD

The Flagship Publication of IEEE COMPUTER SOCIETY Santa Clara Valley Chapter

Volume 2, Issue 3 From the heart of the Silicon Valley July – Sep 2023

Evolution of Convolutional Neural Network (CNN): Compute vs Memory bandwidth for Edge AI

Artificial Intelligence in Test Automation

Building High Performance Modern Web Applications

Upcoming events and conferences:

Slowly changing dimensions and fast changing facts - the story of the traditional data warehouse
Tuesday, August 15, 2023, at 6pm PT

Chapter Open House, Award Ceremony and talk on AI and Conversational Commerce
Tuesday, September 5th, 2023 at 6:30 pm

Panel Discussion on Ethical AI: Shaping the Future Responsibly
Tuesday, Oct 3, 2023, from 11am PT onwards

Website & Media
https://r6.ieee.org/scv-cs/
https://www.linkedin.com/company/78437763/
https://www.linkedin.com/groups/260 6895/
https://www.facebook.com/IEEEComputerSocSCVchapter
https://twitter.com/IEEEComputerSoc

Mailing List
http://listserv.ieee.org/cgi-bin/wa?SUBED1=cs-chap-scv&A=1

Please note:

Feedforward is published quarterly by the Santa Clara Valley (SCV) of the IEEE Computer Society (CS), a non- profit organization. Views and opinions expressed in Feedforward are those of individual authors, contributors and advertisers and they may differ from policies and official statements of IEEE CS SCV Chapter. These should not be construed as legal or professional advice. The IEEE CS SCV Chapter, the publisher, the editor and the contributors are not responsible for any decisions taken by readers on the basis of these views and opinions. Although every care is being taken to ensure genuineness of the writings in this publication, Feedforward does not attest to the originality of the respective authors' content.

Dear Readers,

From the Editor's Desk

Welcome to the third edition of volume 2 of Feedforward, the flagship publication of the IEEE Computer Society, Santa Clara Valley chapter.

In this volume, we present a diverse and compelling collection of articles that delve into some of the most exciting and impactful topics in the tech industry. Join us on a journey through the "Evolution of Convolutional Neural Network (CNN): Compute vs Memory Bandwidth for Edge AI," authored by Dwith Chenna, a seasoned Senior Embedded DSP Engineer with expertise in Computer Vision. Discover the critical relationship between CNNs and compute power, as well as memory bandwidth in the realm of Edge AI.

Additionally, esteemed Technical Leaders at Cisco Systems, Sudipta Debnath, and Debasish Bhadra, present "Artificial Intelligence in Test Automation: A Game-Changer for Green Software Coding," unveiling a revolutionary no-code approach to automation testing that not only accelerates the process but also promotes eco-friendly software development practices.

Moreover, gain valuable insights from Ruchi Agarwal, a skilled Senior Software Engineer at Netflix Inc., in her article "Building High-Performance Modern Web Applications." She shares essential principles and strategies from her experience at the forefront of web development, providing cutting-edge perspectives for industry experts.

This volume promises to inspire and inform readers, leaving them enlightened with new perspectives and ideas.

As we eagerly approach the quarter ahead, we have a host of upcoming events that are set to ignite your passion for technology and innovation. For in-depth information about each event, we invite you to refer to the magazine's pages. Join us on this exciting journey of discovery and advancement!

As always, we extend a warm invitation to join our mailing list, follow us on social media, and actively participate in our events. Together, we will continue to foster networking opportunities and provide valuable resources to our members, propelling our chapter forward and creating a bright future for the technology community.

Thank you for your unwavering support. We hope you thoroughly enjoy reading this edition of Feedforward and look forward to engaging with you throughout this exhilarating quarter.

Submit Articles
https://r6.ieee.org/scv-cs/?p=2036

Stay updated, of upcoming events.
https://r6.ieee.org/scv-cs/category/upcoming-events/

View past events on IEEE.tv and on YouTube
https://ieeetv.ieee.org/search?search_q=scv-cs
https://www.youtube.com/playlist?list=PLLsxQYv4DdJlYcGPwqUJsnHmfqMtB3eSJ

With every best wish,

Meenakshi Jindal San Jose, California, USA
Monday, August 7, 2023

Evolution of Convolutional Neural Network (CNN): Compute vs Memory bandwidth for Edge AI

Dwith Chenna, Senior Embedded DSP Engineer, Computer Vision
MagicLeap Inc. dchenna@magicleap.com

Abstract—Convolutional Neural Networks (CNNs) have greatly influenced the field of Embedded Vision and Edge Artificial Intelligence (AI), enabling powerful machine learning capabilities on resource-constrained devices. This article explores the relationship between CNN compute requirements and memory bandwidth in the context of Edge AI. We delve into the historical progression of CNN architectures, from the early pioneering models to the current state-of-the-art designs, highlighting the advancements in compute-intensive operations. We examine the impact of increasing model complexity on both computational requirements and memory access patterns. The paper presents a comparison analysis of the evolving trade-off between compute demands and memory bandwidth requirements in CNNs. This analysis provides insights into designing efficient architectures and potential hardware accelerators in enhancing CNN performance on edge devices.

Keywords: Convolutional Neural Network (CNN), Network Architecture, Memory Bandwidth, Edge AI

INTRODUCTION

Compute requirements of Artificial Intelligence (AI) models in computer vision (CV) has been increasing rapidly every year. This leads to hardware accelerators focused on computation, usually at the expense of removing other parts such as memory hierarchy. Many computes intense CNN applications have memory bandwidth communication as the bottleneck, in such accelerators. The DRAM memory accelerator scaling has been modest at 2-3x every 2 years, compared to the compute capabilities. The memory requirements for inference are usually much larger than the number of parameters. This is mainly due to the intermediate activation that requires 3-4x more memory, to move the data from/to on-chip local memory. These data transfer limitations can be between on-chip memory and DRAM memory or across different processors, where the bandwidth has been lagging significantly compared to the compute capabilities. As shown in Fig. 1. highlights the increase in compute capacity is 1500x when compared to the DRAM/interconnect bandwidth over the decade. These challenges are commonly referred to as the “memory wall” problem [8], which address both memory capacity and bandwidth.

Fundamental challenges of increasing DRAM/interconnect bandwidth [6], are difficult to overcome. This is intended to only increase the gap between compute and bandwidth capability, making it more challenging to deploy SOTA CNN models at the edge. Addressing this issue needs to rethink the design CNN models, instead of simple scaling schemes based only on FLOPs or compute. The developments in hardware accelerators have been mainly focused on peak compute with limitations of the memory-bound bottleneck. This led to many CNN models that are bandwidth bound, resulting in inefficient utilization of these accelerators.

Model optimization techniques like pruning or quantization enable compressing these models for inference. Pruning is removal of redundant parameters in the model with minimal impact on accuracy. It is possible to prune up to 30% of the model with structured sparsity and 80% through unstructured sparsity with minimal impact of accuracy [15]. This is heavily dependent on the model architecture and any higher pruning results in significant accuracy degradation. Alternatively, quantization approaches have shown successful results in model compression. Quantization refers to the process of reducing the precision of the CNN models to FP16/INT8 or even low bit precision, which leads to significant reduction in memory footprint, bandwidth and latency [16]. INT8 quantization have shown successful results and are adopted by many popular open-source frameworks like tflite/pytorch. Due to the fundamental challenges in increasing compute and memory bandwidth, it is imperative that we rethink the architecture design and deployment of CNN models to handle these limitations. It is possible to sacrifice compute for better bandwidth performance resulting in models that led themselves to efficient deployment on the edge.

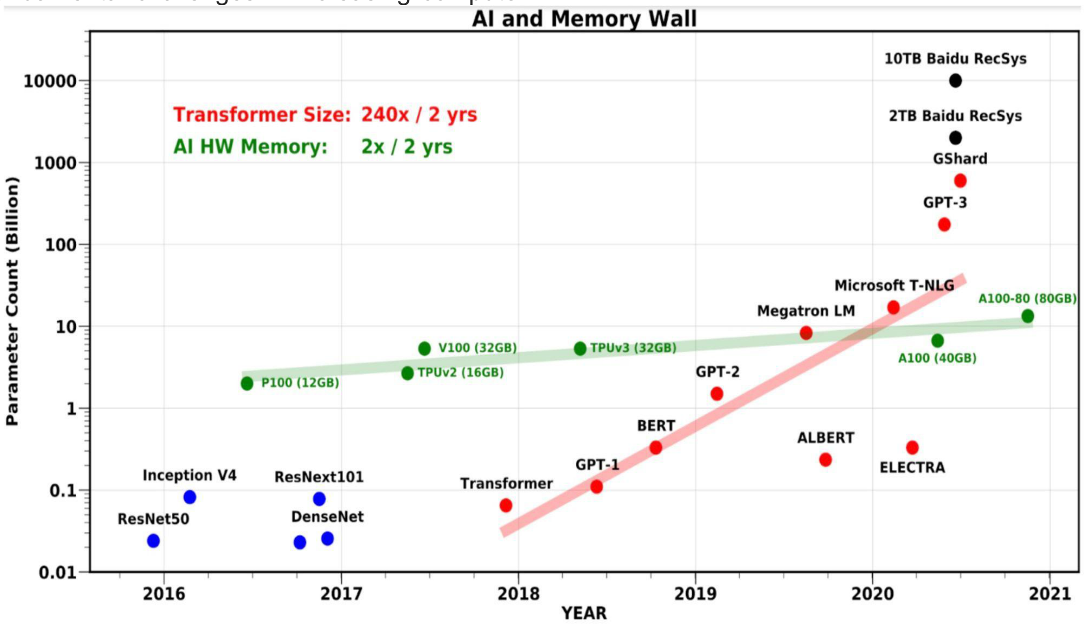


Fig. 1. Scaling of computer and memory bandwidth for AI models [8]

In this paper, we will discuss the evolution of CNN architecture in Section 2.1, highlighting its implications on the compute and memory bandwidth. Section 2.2 and 2.3 we will discuss the underlying assumption for compute and memory bandwidth estimation. Section 3 will discuss the implementation of ONNX tools and analysis of results on a variety of CNN models.

CONVOLUTIONAL NEURAL NETWORKS

Convolutional Neural Networks (CNN) have shown state of the art performance on several computer and Image processing challenges and datasets [1]. This led to CNNs being widely used in applications like Image classification, super resolution, segmentation and object detection. CNN has the ability to extract features at multiple stages, which allows them to learn data patterns especially for computer vision/image processing applications. Many experiments on using different activations, loss functions, operations, parameter optimization have been explored, which lead to increased representation capabilities through architecture innovations. CNN and their application in computer vision/image processing applications allows ideas of spatial, channel, depth of architecture, and multipath processing of information. Multilayer structure of CNN gives it ability to extract low and high-level features. These features can be transferred to generic tasks through Transfer Learning (TL) [1]. Many innovations in CNNs come from different aspects of network parameters, processing units, connectivity layers and optimization strategies. Recent developments in hardware technologies also allowed developing and deploying these models, making them ubiquitous in various applications running in cloud to edge applications.

In this section we focus on evolution of CNN architectures with emphasis on compute and memory bandwidth, allowing a much better understanding on the network architecture impact on deployment of the edge.

AlexNet

AlexNet, proposed by Krizhevsky et al., marked a significant advancement in Convolutional Neural Network (CNN) architectures for image classification and recognition tasks. It surpassed traditional methods by employing a deeper model and strategic parameter optimization [7]. To address hardware limitations, the model was trained on dual NVIDIA GTX 580 GPUs, allowing it to extend to 8 layers. This expansion enhanced its capacity to learn from diverse image categories and improved its adaptability to varying resolutions. Yet, deeper architectures brought the challenge of overfitting. With 60 million parameters, overfitting was mitigated through measures like dropout layers and data augmentation. The model's prowess was evident when it clinched the 2012 ImageNet competition, outperforming its closest competitor by a significant 11% in error reduction.

VGG

The success of CNNs in image recognition spurred architectural advancements. Simonyan et al.'s creation, VGG-16, employed simple yet effective principles [9] to enable deeper CNN models. This marked a significant leap in deep learning and computer vision, introducing the era of very deep CNNs. VGG-16 innovated by replacing larger filters with a compact 3x3 variant, showcasing that stacking smaller filters could replicate larger ones. This approach lowered computational complexity, reducing parameters and model size. The model's simplicity, depth, and uniform structure made it a frontrunner, securing 2nd place in the 2014 ILSVRC competition. However, its 138 million parameters posed computational challenges, limiting its deployment on edge devices.

Inception

Inception-V1, also known as GoogleNet, emerged as the victor of the 2014 ILSVRC competition with a focus on high accuracy and reduced computational complexity [10]. Named after its Inception blocks, the model employed multi-scale convolution transforms via split, transform, and merge techniques. These blocks encompassed filters of varying sizes (1x1, 3x3, and 5x4) to capture diverse spatial information, effectively addressing image category diversity at different resolutions. Computational efficiency was maintained through 1x1 convolution bottleneck layers preceding larger kernels, and a global average pool replaced the fully connected layer to reduce connection density. By applying such adjustments, parameters were pruned from 138 million to 4 million. This inception concept evolved in subsequent versions like Inception-V2, V3, V4, and Inception-ResNet, which introduced asymmetric filters to further streamline compute complexity.

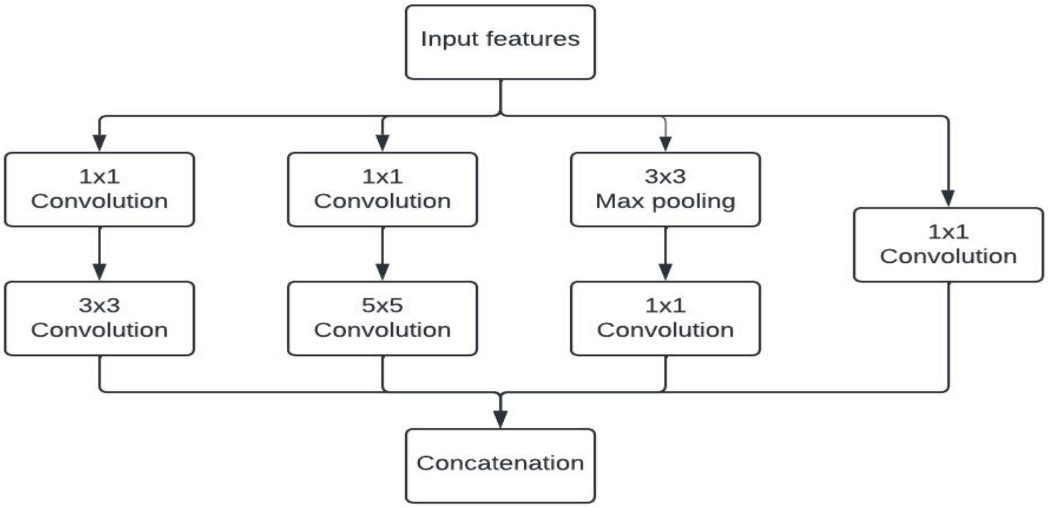


Fig. 2. Inception block showing the split compute and merge.

ResNet

ResNet, devised by He et al. [5], brought a revolutionary shift in CNN architecture by introducing residual learning and efficient techniques for deep network training. It tackled the vanishing gradient issue, enabling even deeper CNN models. ResNet's innovation allowed a 152-layer deep CNN that triumphed in the 2015 ILSVRC competition. Compared to AlexNet and VGG, ResNet's 20x and 8x depth respectively came with relatively lower computational complexity. Empirical findings favored ResNet models with 50/101/152 layers over shallower variants. It showcased remarkable accuracy enhancements for complex visual tasks like image recognition and localization on the COCO dataset. ResNeXt [17] emerged as an improvement, treating it as an ensemble of smaller networks. Utilizing diverse convolutions (1x1, 3x3, 5x5) appended with 1x1 bottleneck convolution blocks, ResNeXt explored various topologies across different paths.

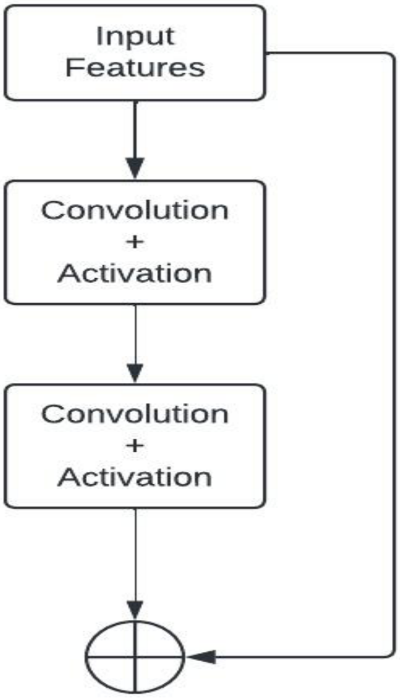


Fig. 3. Residual block structure used in ResNet

DenseNet

DenseNet capitalizes on the insight that close connections between input and output lead to deeper, accurate, and efficient CNN models [18]. By linking each layer to the following layer in a feed-forward manner, DenseNet incorporates feature maps from all preceding layers through concatenation. This strategy enables the model to discern added and preserved information within the network. With a reasonable parameter count, the model utilizes a relatively smaller number of feature maps. Key attributes include significant parameter reduction, promotion of feature reuse, and handling of vanishing gradients. Impressively, DenseNet achieves performance on par with ResNet while demanding fewer parameters and computational resources.

MobileNet

MobileNets [19] leverage depthwise convolutions to drastically decrease model size and computational demands. Engineered for resource-limited mobile devices with latency-sensitive tasks, the architecture allows flexibility through width and resolution multipliers. This empowers users to balance computation and latency as needed. The depthwise convolution based MobileNet reduces parameters from 29.3M to 4M and FLOPs by nearly 10x. These models, exemplified by MobileNet, achieve leaner and deeper structures, fostering representation diversity and computational efficiency. This design aligns well with cache-based systems and suits bandwidth-constrained edge AI applications due to their compact yet comprehensive nature.

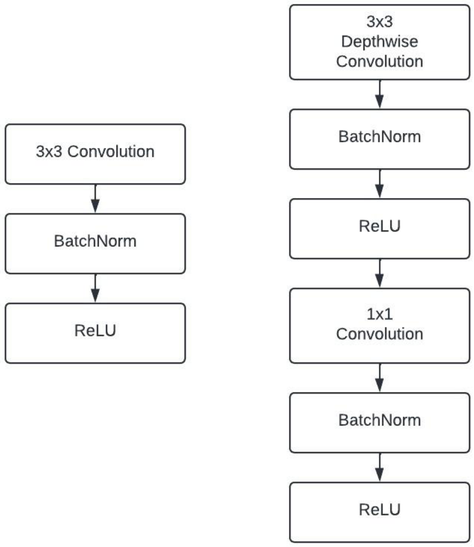


Fig. 4. Convolution layer with batchnorm and activation, depth wise convolution with 1x1 convolution layers

SqueezeNet

SqueezeNet stands out with its small footprint, boasting 50x fewer parameters yet achieving AlexNet-level accuracy on ImageNet [20]. Designed to enhance training communication and operate on resource-constrained hardware, it optimizes memory utilization. SqueezeNet's compactness arises from substituting 3x3 filters with 1x1 filters, resulting in a 9x parameter reduction. By employing squeeze layers, channel count reduction in 3x3 convolutions is utilized to maintain efficiency. Late stage downsampling, known as delayed downsampling [14], ensures higher accuracy is maintained.

EfficientNet

EfficientNet, developed through AutoML NAS [21], strategically optimizes accuracy and compute efficiency. Employing mobile inverted residual bottleneck convolutions (MBConv) akin to MobileNet, it leverages compound scaling [22] for diverse networks across computation budgets and model sizes. EfficientNet excels in accuracy and efficiency over existing CNNs, substantially reducing model size and FLOPs. For instance, EfficientNet-B0 surpasses ResNet-50's performance with a 5x parameter reduction and 10x FLOP reduction. These models outperform counterparts like ResNet, DenseNet, and Inception while employing significantly fewer parameters.

Table 1 summarizes the accuracy of these models on the ImageNet classification accuracy task. The overview of ImageNet accuracy across various network architectures offers insightful observations. Pioneers like AlexNet achieved commendable accuracy (57.2% top-1, 80.2% top-5), setting the foundation. VGG16 and VGG19 maintained steady performance at 71.5% top-1 accuracy. Inception models, like Inception V2 and GoogleNet, harnessed innovation for strong accuracy (up to 73.9%). The ResNet series pushed boundaries with deep structures, reaching a remarkable 78.3% top-1 accuracy. DenseNet's feature reuse led to 74.9% accuracy. Models like SqueezeNet (57.5%) optimized compactness, while MobileNet V2 (71.8%) and ShuffleNet V2 (67.9%) catered to mobile devices. EfficientNet achieved efficiency and accuracy (77.1%). Balancing accuracy, size, and computation remains crucial for diverse applications and hardware constraints.

Model	Model Size	Top 1%	Top 5%
AlexNet	238 MB	54.80	78.23
VGG16	527.8 MB	72.62	91.14
VGG19	508.5 MB	73.72	91.58
Inception V1	27 MB	67.23	89.6
Inception V2	44 MB	73.9	91.8
GoogleNet	27 MB	67.78	88.34
ResNet18	44.7 MB	69.93	89.29
ResNet34	83.3 MB	73.73	91.40
ResNet50	97.8 MB	74.93	92.38
ResNet101	170.6 MB	76.48	93.20
ResNet152	230.6 MB	77.11	93.61
DenseNet	32 MB	60.96	92.2
SqueezeNet	5 MB	56.85	79.87
MobileNet V2	13.3 MB	69.48	89.26
ShuffleNet V2	8.79MB	66.35	86.57
EfficientNet	51.9 MB	80.4	93.6

Table 1. Summary of Image classification accuracy results for different mode

COMPUTE

As CNN architectures become deeper and more complex, it's essential to estimate their computational requirements accurately. Two commonly used metrics for estimating computational capacity are Floating Point Operations per Second (FLOPs) and Multiply-Accumulate (MAC) operations. FLOPs provide an estimate of the total number of floating-point operations (additions and multiplications) required to perform the forward pass of a neural network. For CNNs, this includes operations like convolutions, pooling, element wise addition/multiplication and fully connected layers. MAC operations are a fundamental building block of many mathematical computations, especially in matrix multiplications. In the context of CNNs, MAC operations occur during convolutional and fully connected layers. For operations like convolution and fully connected layers, we can estimate FLOPs and MAC through

$FLOP \simeq 2 * MAC$

In this section we estimate the compute complexity of the CNN through FLOPs, which can better estimate operations like elementwise add/multiply, average pooling which are found more frequently in many of these modern architectures. Estimation of FLOPs for different architectures is dependent on the different operations used in the network architecture. We will discuss widely used operations and how to estimate the FLOPs for these operations. Due to the simplified structure and scaling of the deep CNNs model we tend to see many repetitive operations across different network architecture. This makes it easy to estimate the compute complexity by analyzing different operations used to form the layer of the network. This section section we will review five major types of operations and estimate FLOPs for these operations.

Convolution

Convolutions is the fundamental building block for CNN models. It leverages the spatial sparsity to reduce the number of operations and still be effective in image processing/computer vision applications. It usually consists of multi-dimensional input and weight that generate the output feature maps as shown in Fig. 5.

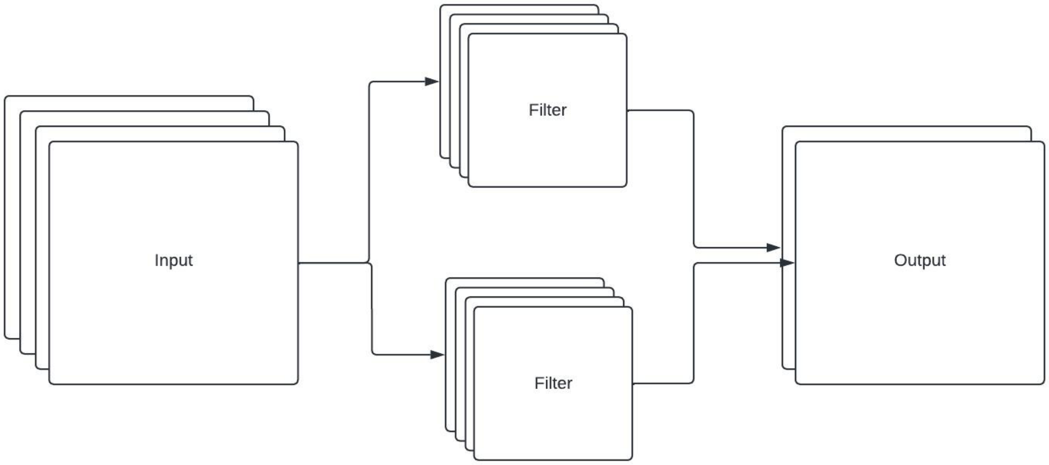


Fig. 5. Convolution operation with multiple filters

The convolutions operations can be represented through the equation (1), we can estimate the FLOPs based on the input, weight and output dimensions.

$$y_{out}^{i,k} = \sum_{i,j=1}^{M} \quad x_{in}^{i,j} w^{i,j} + b^{k} \qquad (1)$$

flops = C_in * H_out * W_out * C_out * H_k * W_k / (stride * stride * group) (2)

Where C_in is input channel dimension, H/W/C_out are output dimensions, H/W_k kernel dimensions, stride and group.

Fully Connected (FC)

Fully connected layers are usually seen at the head of the network. Due to their extensive connectivity they tend to have orders of magnitude compute that increase with the fixes. In most cases the feature map sizes are reduced to avoid such bottlenecks in the network.

flops = N_in * H_w * W_w (3)

Average Pool

Spatial dimensions of the feature maps are reduced as we go deeper into the network, allowing the network to capture high spatial features. This is achieved by using pooling operations that reduce the spatial dimension with fixed kernel size and stride. Average or max pooling operations are widely used in CNNs. In case of max pooling operations, the computation is negligible as it picks the max from the kernel dimensions. In this section we will look at estimating compute for average pooling operation.

flops = C_in * H_out * W_out * H_k * W_k / stride (4)

Where C_in is input channel dimension, H/W_out are output feature map dimensions, H/W_k are the pooling kernel dimensions and stride.

Elementwise Add/Mul

Elementwise Add/Multiply operations are prevalent in many modern network architectures after ResNet architecture showed promising results to vanishing gradient problem. This enabled deeper networks with larger representation capacity. The compute estimate for these operations is through eq (5), where C_in are input feature dimensions, H/W_in are input spatial dimensions.

flops = C_in * H_in * W_in (5)

MEMORY BANDWIDTH

Memory bandwidth is a critical consideration when deploying Convolutional Neural Network (CNN) models for inference on hardware platforms. CNNs involve intensive data movement between memory and processing units, making memory bandwidth a potential bottleneck for performance. Estimating memory bandwidth requirements helps in optimizing model deployment and selecting appropriate hardware. Memory bandwidth refers to the rate at which data can be read from or written to memory. In the context of CNN inference, memory bandwidth is crucial because it involves data movement that needs to be moved between memory and processing unit. Two main components of data movement include i) Weight Loading, CNN layers utilize learned weights that are stored in memory. Fetching these weights efficiently is vital for smooth inference ii) Feature Maps, Intermediate feature maps generated during computation are stored in memory for subsequent layers' use. Efficient access to these maps impacts inference speed.

Edge AI devices usually remove memory hierarchy in favor of larger compute, small silicon area and power. These systems have dedicated I/O control blocks that allow instruction/data to move independently. Independent instruction/data buses allow users to hide the memory and compute latency through a double buffer, which increases the overall throughput. These architectures are memory and power efficient making them a popular choice for resource constrained hardware with strict latency and power budget.

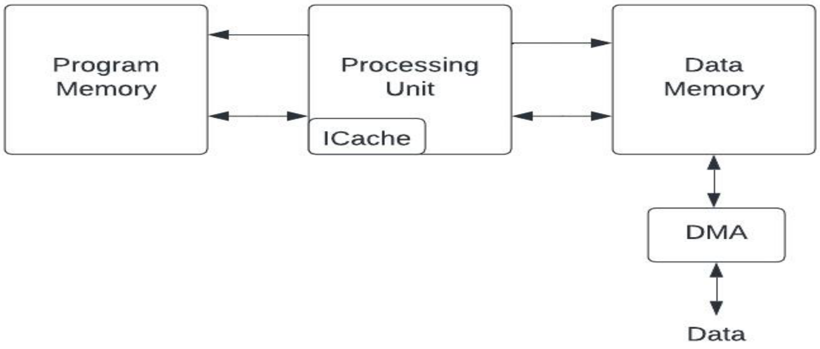


Fig. 6. Super Harvard Architecture improvement include instruction cache and dedicated DMA controller.

In order to estimate bandwidth, we need to move the intermediate activation to and from the local memory to system memory. This puts extra burden on the bandwidth requirement, when estimating bandwidth, we need to consider the data from system memory to local memory and back, resulting in twice the memory movement. These estimates are made per model inference, by calculating the intermediate activation and weights that need to be moved in and out of local and system memory.

IMPLEMENTATION

In order to estimate the computer and memory bandwidth requirements, based on discussion we use ONNX models. Open Neural Network Exchange (ONNX) [14], is an open-source machine independent format, widely used for exchanging neural network models. We infer the sizes of intermediate results for the ONNX model in order to estimate the bandwidth requirement. For bandwidth estimation we assume model optimization techniques batchnorm folding and layer fusion, which fuses batch normalization and activation functions into the previous convolution operation.

RESULTS

In this section, we will review the results of compute and memory estimation. These estimates are based on a group of CNN models that represent the evolution of CNN architectures throughout the past few years. For this analysis we used AlexNet[7], VGG-16[9], InceptionV1/V2[10], ResNet18-152[5], DenseNet[18], MobileNet[19], SqueezeNet[20] and EfficientNet[22], as discussed in Section 2. Fig. 7. shows network vs compute (FLOPs), the FLOPs are in log scale to accommodate a variety of networks. All these ONNX models [] use standard input sizes of 224x224x3, which enables us to do a fair comparison across different architectures.

`

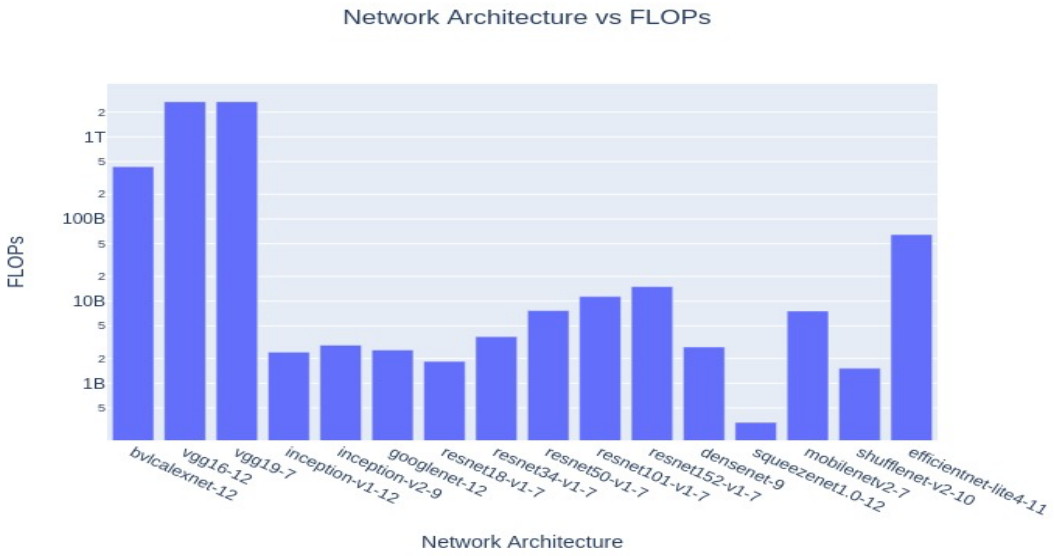


Fig. 7. Network architecture vs Compute capacity (FLOPs)

We clearly see a trend towards small FLOPs, as the network architecture matures with different architecture optimization techniques. However, bandwidth doesn't show any consistent trend, it is from the fact that many of these architecture designs do not consider the bandwidth constraint of the edge AI applications.

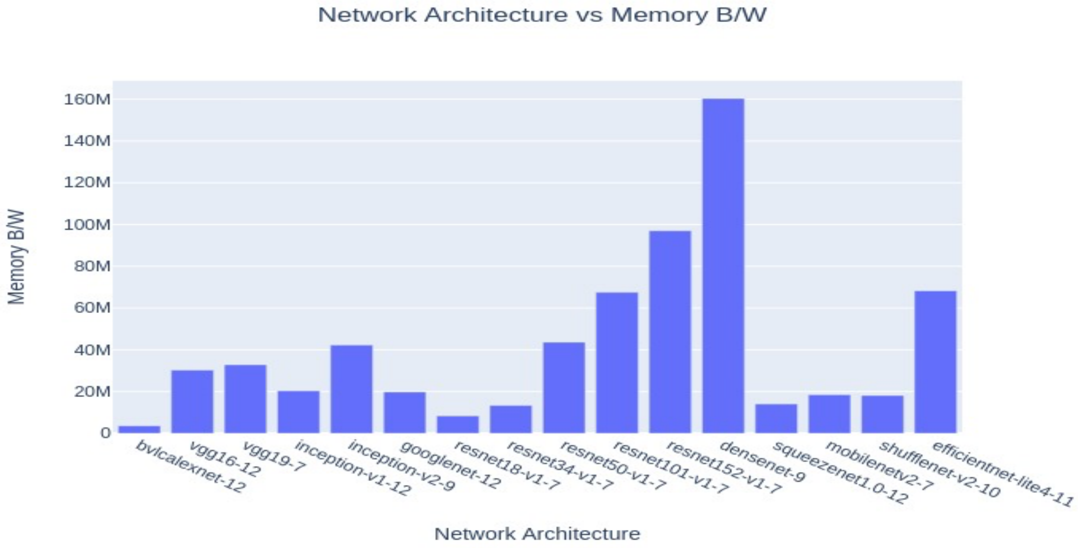


Fig. 8. Network architecture vs Memory Bandwidth (/Inference)

A comparison of the compute vs memory bandwidth (Fig. 9), shows that mobilenet networks designed for mobile application tend to be leaner and deeper, requiring higher bandwidth, might not be optimal architecture for the edge AI application. SqueezeNet shows superior performance in terms of both compute and memory bandwidth tradeoff, as it was designed for smaller size and minimal memory bandwidth.

SUMMARY

This paper provides a comprehensive analysis of the evolution of CNN compute requirements and memory bandwidth in Edge AI applications. The rapid evolution of CNNs models, using FLOPs/parameters as estimates of performance results in suboptimal models. Especially hardware bandwidth constraints act as bottlenecks to model performance. It is essential to estimate both compute and bandwidth to design architecture that provides optimal performance for Edge AI application, with strict constraints on latency, bandwidth and power. In this paper, we offer valuable insights to researchers, practitioners, and hardware designers, facilitating a deeper understanding of the trade-offs involved in optimizing CNNs for edge deployment.

CONFLICT OF INTEREST

The author of the manuscript hereby declares that we have no conflicts of interest to disclose related to the research work presented in this manuscript. I confirm that no financial, personal, or professional relationships or affiliations exist that could be perceived as a conflict of interest with regards to the research presented in the manuscript. I further confirm that no funding sources or sponsors played a role in the design, execution, analysis, or interpretation of the study or in the preparation of this manuscript.

DATA AVAILABILITY

The datasets generated during and/or analyzed during the current study are available in the github repository: https://github.com/onnx/models/tree/main/vision/classification.

The tools used for the analysis will be made available in github repository: https://github.com/cyndwith/Evolution-of-Convolutional-Neural-Network-CNN-Compute-vs-Memory-bandwidth-for-Edge-AI

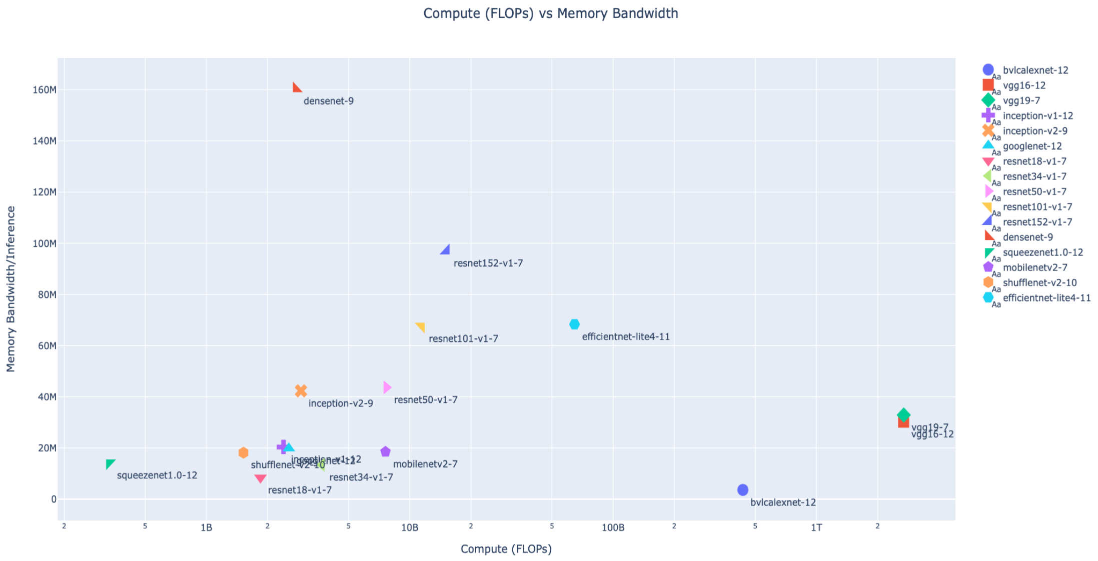


Fig. 9. Computer (FLOPs) vs Memory Bandwidth (/Inference)

REFERENCES

[1] Qiang Yang, Pan SJ, Yang Q, Fellow QY (2008) A Survey on Transfer Learning. IEEE Trans Knowl Data Eng 1:1–15. doi: 10.1109/TKDE.2009.191

[2] Zeiler MD, Fergus R (2013) Visualizing and Understanding Convolutional Networks. arXiv Prepr arXiv13112901v3 30:225–231. doi: 10.1111/j.1475-4932.1954.tb03086.x

[3] Simonyan K, Vedaldi A, Zisserman A (2013) Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps. 1–8. Doi: 10.1080/00994480.2000.10748487

[4] Szegedy C, Vanhoucke V, Ioffe S, et al (2016b) Rethinking the Inception Architecture for Computer Vision. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition. IEEE, pp 2818–2826

[5] He K, Zhang X, Ren S, Sun J (2015a) Deep Residual Learning for Image Recognition. Multimed Tools Appl 77:10437–10453. doi: 10.1007/s11042-017-4440-4

[6] Patterson DA. Latency lags bandwidth. Communications of the ACM. 2004 Oct 1;47(10):71–5.

[7] Krizhevsky, A.; Sutskever, I.; Hinton, G.E. Imagenet classification with deep convolutional neural networks. In Proceedings of the Advances in Neural Information Processing Systems, Lake Tahoe, NV, USA, 3–6 December 2012; pp. 1097–1105.

[8] https://medium.com/riselab/ai-and-memory-wall-2cb4265cb0b8

[9] Karen Simonyan and Andrew Zisserman. 2015. Very deep convolutional networks for large-scale image recognition. In ICLR.

[10] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, “Going deeper with convolutions,” in Proc. IEEE Conference on Computer Vision and Pattern Recognition, Boston, Jun. 2015, pp. 1–9.

[11] Dwith Chenna, Fixed Point Implementation of Convolutional Neural Networks (CNN) on Digital Signal Processor (DSP), International Journal of Artificial Intelligence & Machine Learning (IJAIML), 2(1), 2023, pp. 23-34.

[12]Github:https://github.com/cyndwith/Evolution-of-Convolutional-Neural-Network-CNN-Compute-vs-Memory-bandwidth-for-Edge-AI

[13] Bai, J., Lu, F., Zhang, K. et al.: ONNX: Open Neural Network Exchange, GitHub (online), available from https://github.com/onnx/onnxi (accessed 2020-07-01).

[14] Kaiming He and Jian Sun. Convolutional neural networks at constrained time cost. In CVPR, 2015

[15] Baoyuan Liu, Min Wang, Hassan Foroosh, Marshall Tappen, Marianna Pensky; Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 806-814

[16] Jiaxiang Wu, Cong Leng, Yuhang Wang, Qinghao Hu, Jian Cheng; Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 4820-4828

[17] Hitawala, S. Evaluating ResNeXt Model Architecture for Image Classification. arXiv 2018, arXiv:1805.08700.

[18] F. Iandola, M. Moskewicz, S. Karayev, R. Girshick, T. Darrell, and K. Keutzer, “Densenet: Implementing efficient convnet descriptor pyramids,” arXiv preprint arXiv:1404.1869, 2014.

[19] A. G. Howard, M. Zhu,. B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto and H. Adam. MobileNet : Efficient Convolutional Neural Networks for Mobile Applications. arXiv preprint arXiv:1704.04861,2017.

[20] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, “SqueezeNet: Alexnet-level accuracy with 50x fewer parameters and < 0.5 MB model size,” in ICLR, 2016.

[21] X. He, K. Zhao, and X. Chu, “AutoML: A survey of the state-of-theart,” Knowl.-Based Syst., vol. 212, Jan. 2021, Art. no. 106622.

[22] M. Tan and Q. V. Le, “EfficientNet: Rethinking model scaling for convolutional neural networks,” in ICML, 2019.

[23]Github: https://github.com/onnx/models/tree/main/vision/classification

Artificial Intelligence in Test Automation

A game changer in automation testing – no code approach towards green software coding

Sudipta Debnath, Technical Leader Cisco Systems, Inc. North Carolina, 27560, USA

Debasish Bhadra, technical Leader Cisco Systems India Pvt. Ltd. Bangalore, 560103, INDIA

Abstract—With Agile practices, software development lifecycles are becoming more complicated. As delivery time spans reduce, testers need to impart feedback and evaluations instantly to development teams. In the recent era of DevOps, the focus is on implementing a continuous testing approach to provide full visibility of product quality in a smarter way. Still, testing is always a challenging situation. While Test automation, continuous testing, and the low-code/no-code approach are trying to handle the testing process smarter and faster, it is quite evident that the key to streamlining software testing and making it smarter and more efficient is artificial intelligence. Today, Artificial Intelligence and machine learning are centered on training software to understand input data versus output. This is very similar to the testing activities performed manually. A tester provides an input into a field and looks for an expected output. With the introduction of Artificial Intelligence models, the machine comes up with all testing possibilities and automatically optimizes the test case creation for the testing process. It even handles changes to the automation code and test cases that previously had to be made by the manual effort of QA professionals. This paper aims to highlight how the Artificial Intelligence model can be a game-changer in the testing industry with a zero-touch human model, proven to automate automation.

In any agile development environment, Test Automation is a rage. It is a known fact, or one can say it is a myth that automation works well only for regression or smoke tests. While automation ensures speedy deliverables and reduces costs in the testing lifecycle, it is crucial for any organization to understand the exact intentions behind each test case planned for a release, deliverables, or platform.

To assess the situation from a delivery perspective, software teams are under constant pressure to deliver better quality products in shorter timeframes. To achieve that, testing has shifted both left and right, and the automation of tests plays a crucial role. However, traditional test automation has become a bottleneck, as it requires domain-specific knowledge. There is a need for individuals who not only know how to automate but also possess skills in analyzing and understanding complex data structures, statistics, and algorithms.

At present, there are no automation solutions that can automatically predict what to test, how to test, and with zero human intervention can design a test suite, execute it, and publish results considering all probable combinations required for testing towards the business goal.

In today's test automation approach, challenges are as follows:

- What to automate and how to predict it automatically.
- How to decide on the right test suite.
- Making test automation reusable without constant learning efforts.
- Test script generation is not fully automated and requires manual intervention.
- Test execution is automated, but generating test workflows is still manual.

But before we start using Artificial Intelligence as a model to solve all these testing challenges, let us understand "What is Artificial Intelligence" in the following section and how it can make it possible to automate an automation.

ARTIFICAL INTELLIGENCE

95% of today's businesses have adopted cloud-native automation. Statistics show that only one-third can successfully achieve the anticipated ROI (Return on Investment). While the cloud aims to set up next-level computing power and provide access to new kinds of data in the right quantity and quality, Artificial Intelligence (AI) serves as the bridge to convert that data into business value.

Artificial Intelligence is most often treated as a complex, confusing, and misunderstood term. Testing organizations are often unaware of the role that Artificial Intelligence can play in common activities, experiences, and interactions today. How did Artificial Intelligence become so pervasive, and what is its future? To understand and address all these questions, we must first comprehend what Artificial Intelligence is.

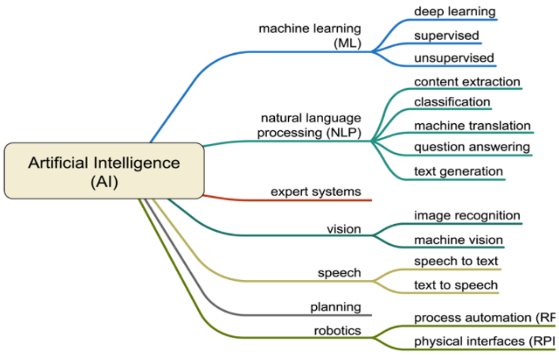
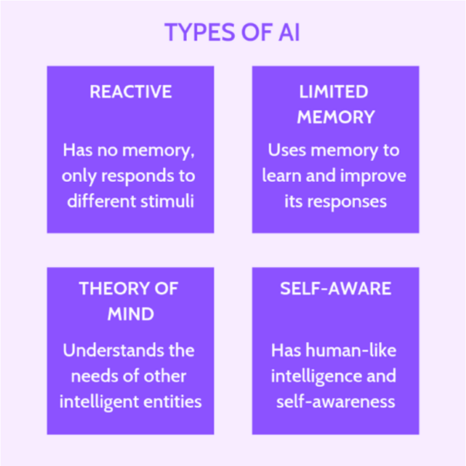


FIGURE 1. What is Artificial Intelligence

Artificial intelligence is a process that simulates human intelligence processes by machines, especially computer systems. Specific applications of Artificial Intelligence include expert systems, natural language processing, speech recognition, and machine vision. According to Dr. Peter Norvig, a Berkeley professor and director of Google Research, 'Artificial Intelligence is all about figuring out what to do when you don't know what to do. Regular programming is about writing instructions for the computer to do what you want it to do when you do know what you want it to do. Artificial Intelligence is for when you don't.' (Ref: Artificial Intelligence: A Modern Approach, US 4th Edition by Stuart Russell and Peter Norvig)

In an article published in 2016, Arend Hintze, an assistant professor of integrative biology and computer science and engineering at Michigan State University, explained that Artificial Intelligence can be categorized into four types. His explanation reveals a fact that starting with task-specific intelligent systems in wide use today and progressing to sentient systems, which do not yet exist, the categories can be simplified as follows:



TYPES OF AI

REACTIVE

Has no memory, only responds to different stimuli

LIMITED MEMORY

Uses memory to learn and improve its responses

THEORY OF MIND

Understands the needs of other intelligent entities

SELF-AWARE

Has human-like intelligence and self-awareness

FIGURE 2. Different types of Artificial Intelligence

- Type 1: Reactive machines - These Artificial Intelligence systems have no memory and are task-specific. For example, a program that can identify actions and make predictions can be considered in this category. However, it can't make informed future decisions as it doesn't have memory that can be used to define a steady pattern based on past experiences. To clarify further, reactive machines excel at performing specific tasks and responding to specific inputs in real-time. They don't retain any information from past interactions, which limits their ability to adapt or learn from experience. As a result, they lack the capability to make decisions or predictions based on historical data or patterns.
- Type 2: Limited memory - These Artificial Intelligence systems have memory, allowing them to use past experiences to inform future decisions. For example, some decision-making functions in self-driving cars are designed this way. They can learn from previous situations to improve their performance and adapt to varying conditions.

- Type 3: Theory of mind - While this term originates from psychology, in the context of Artificial Intelligence, it implies that the system possesses social intelligence to understand emotions. This type of Artificial Intelligence will be able to comprehend human intentions and predict behavior based on emotional cues and interactions.

- Type 4: Self-Awareness - In this category, Artificial Intelligence systems have a sense of self, granting them consciousness. Machines with self-awareness understand their own current state, although such systems do not yet exist in today's world. This type of AI is considered a potential future development in the field of Artificial Intelligence.

These four types of Artificial Intelligence represent a progression from simple, task-specific systems to more sophisticated and complex AI that can understand human emotions and even possess a level of consciousness. As technology advances, it remains an exciting area of research and development in the field of Artificial Intelligence.

While this paper acknowledges that Artificial Intelligence is a vast field with numerous applications, the focus will now shift to exploring how Artificial Intelligence models can revolutionize the testing industry. Specifically, we will concentrate on how organizations can leverage Artificial Intelligence in Test Automation to create a matured service offering with assured benefits.

By integrating Artificial Intelligence into Test Automation, organizations can achieve significant advancements in the efficiency, accuracy, and coverage of their testing processes. AI-powered testing models have the potential to identify patterns, anomalies, and critical issues in complex software systems, making them indispensable tools for quality assurance.

One of the key areas where AI can be a game-changer is in test case generation and selection. Traditional testing methodologies often require manually designing test cases, which can be time-consuming and may not cover all possible scenarios. With AI, testing frameworks can automatically generate test cases based on historical data, code analysis, and system behavior, leading to comprehensive test coverage and quicker testing cycles.

Moreover, AI can enhance defect prediction and detection. By analyzing past testing data and application performance, AI models can identify potential areas of concern, enabling testers to proactively address vulnerabilities before they escalate into critical issues. This proactive approach reduces the risk of costly defects and ensures a more reliable software product.

Incorporating AI into test execution and maintenance can also significantly reduce human effort and resources.

AI-powered bots can autonomously execute test cases across various platforms and devices, providing continuous testing coverage and accelerating the release process. Additionally, AI models can learn from test results, continuously improving test scripts and adapting to changes in the application under test.

ARTIFICIAL INTELLIGENCE MAKES INDUSTRY TO MOVE FROM PRACTICE TO PERFORMANCE

It is often said that Test Automation is most suitable for tasks that involve repetitive testing, with regression testing being a top priority for automation. While it is true that Artificial Intelligence (AI) can bring quicker and more efficient results in regression testing, its potential goes beyond this domain. Let's explore other areas where AI can add value in the testing process:

- Test Generation - AI can automatically generate tests using sophisticated test design algorithms, automation code for various test ecosystems, and relevant test data. This streamlines the test creation process, ensuring comprehensive coverage and reducing manual effort.
- Test Execution - AI can dynamically identify screens and elements in any software application and automatically execute test cases, eliminating the need for manual test recording and script improvisation. This approach ensures more reliable and accurate test execution.
- Test Optimization - AI can optimize legacy test assets by converting them into a minimum set of test cases based on business criticality. Additionally, it can analyze customer and production data to identify important features and potential areas for automation, leading to more efficient and effective testing practices.
- Test Analysis - AI models can analyze both code and the tests that run against it to

determine precise test coverage. By continuously comparing test results to test history, AI can dynamically detect changes and regressions, resulting in more stable software releases.

In the testing industry, AI solutions that focus on accuracy, automation, and improved return on investment (ROI) are particularly promising. When AI-driven practices are integrated into the culture of testing, it leads to better performance and efficiency. AI's ability to automate repetitive tasks and optimize testing processes helps in delivering higher quality products with reduced time-to-market.

Through automation and the integration of AI, the best practices of the testing industry can be elevated to a new level of quality and performance. AI-driven testing solutions have the potential to enhance not only regression testing but also the overall testing lifecycle, ultimately leading to improved software reliability and customer satisfaction.

In the following section, we will explore and visualize how these possibilities can be achieved by leveraging AI in the testing process, and the potential impact it can have on the industry.

AUTOMATE AUTOMATION

The terms "Artificial Intelligence" (AI) and "Applied Intelligence" are often used interchangeably, but they can have distinct meanings depending on the context in which they are used. Let's clarify the difference between the two:

Artificial Intelligence (AI): AI refers to the simulation of human intelligence in machines, enabling them to perform tasks that typically require human intelligence. It encompasses various techniques, algorithms, and models that allow machines to learn from data, reason, and make decisions. AI systems can analyze patterns in data, recognize speech, process natural language, and perform tasks that previously required human intervention.

Applied Intelligence: Applied Intelligence, on the other hand, is a specific subset or application of AI. It involves leveraging AI technologies and techniques to address specific business problems or challenges effectively. In other words, Applied Intelligence is the practical implementation and use of AI to derive value and insights from data in specific domains,

In the context of our statement, we are using "Applied Intelligence" to emphasize the practical application of AI in the automation of the test automation process. By using a data-driven intelligent model, we aim to enhance the speed, efficiency, and accuracy of the test automation process. This implementation of AI to drive automation and optimize testing aligns with the concept of Applied Intelligence.

In summary, while AI represents the broader field of simulating human intelligence in machines, Applied Intelligence refers to the practical implementation of AI to solve specific problems and achieve specific objectives in various domains, including test automation. The data-driven intelligent model you are developing is a manifestation of Applied Intelligence, enabling faster and more efficient test automation and making the entire ecosystem smarter and more effective.

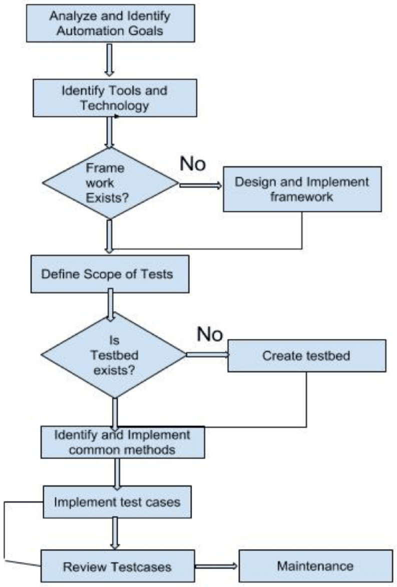

Analyze and Identify Automation Goals
Identify Tools and Technology
Frame work Exists?
No
Design and Implement framework
Define Scope of Tests
Is Testbed exists?
No
Create testbed
Identify and Implement common methods
Implement test cases
Review Testcases
Maintenance

FIGURE 3. Traditional Test Automation Approach

Hence, the scope of test automation should go beyond just automating the test execution process. Integrating test case generation, selection, and input data set management through a predictive data model can bring significant benefits to the test automation process. As you mentioned, organizations must consider automating the entire test automation process to achieve cost-effectiveness, efficiency, and better outcomes.

The next step in applying intelligence to the test automation process is to shift towards "left shifting" test automation. This means integrating testing activities earlier in the Software Development Life Cycle (SDLC) to detect and address defects at an early stage. By doing so, organizations can identify issues in the development phase, reduce rework, and ensure higher software quality.

In Figure 3, you have visualized the traditional test automation approach, which primarily focuses on test execution at the end of the development cycle. However, by adopting a more intelligent and automated approach, the entire testing process can be enhanced, as shown below:

Automated Test Scope Definition: AI-powered tools can analyze requirements, user stories, and code changes to automatically define the scope of testing. This ensures that all relevant features and functionalities are included in the test suite.

Automated Test Goal Identification: AI can assist in identifying the most critical areas of the application that require testing based on historical data and defect trends. It helps in setting automation goals that align with the business priorities.

Automated Test Case Generation: AI-driven algorithms can automatically generate test cases based on the identified test goals and the defined scope. This accelerates the test design process and ensures comprehensive test coverage.

Automated Input Test Data Generation: AI can generate diverse combinations of input test data, including boundary values, edge cases, and real-world scenarios. This improves test coverage and helps in identifying hidden defects.

By applying intelligence in test automation, organizations can build an AI-defined test automation approach that optimizes testing efforts, reduces manual intervention, and enhances software quality. The process becomes more efficient, and the entire SDLC benefits from faster feedback and reduced time-to-market.

AUTOMATION TEST MODEL

The Automation Test Model (ATM) we described is an excellent example of an Applied Intelligence model that simplifies and enhances the test automation process. It encompasses two crucial components:

Automated Modeling with Reusable Test Assets: The ATM leverages automation to create models that are based on reusable test assets in a specific domain. These test assets may include test cases, test data, test scenarios, and other testing artifacts that are commonly used in the domain. By automating the modeling process, the ATM can quickly generate and adapt test models for different projects, reducing the manual effort required to create new test cases from scratch.

Automated Test Case Generation with Ready-to-Execute Test Scripts: The ATM goes a step further by automating the test case generation process, producing ready-to-execute test scripts. This automation eliminates the need for manual test case creation, reducing the risk of human errors and speeding up the testing process. With automated test scripts readily available, testing teams can execute test cases more efficiently and achieve quicker feedback on the software's performance and quality.

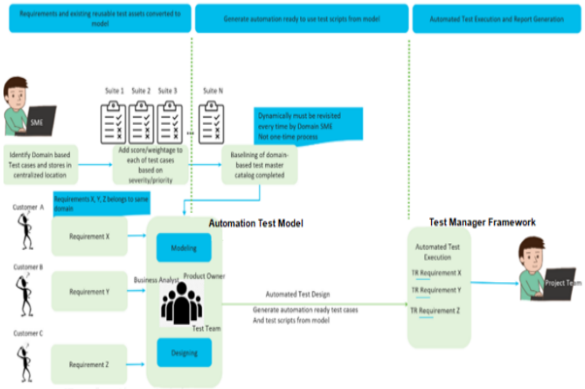

Requirements and existing reusable test assets converted to model
Generate automation ready to use test scripts from model
Automated Test Execution and Report Generation
SME
Suite 1
Suite 2
Suite 3
Suite N
Dynamically must be revisited every time by Domain SME Not one time process
Identify Domain based Test cases and stores in centralized location
Add score/weightage to each of test cases based on severity/priority
Baselining of domain-based test master catalog completed
Customer A
Requirements X, Y, Z belongs to same domain
Customer B
Customer C
Requirement X
Requirement Y
Requirement Z
Automation Test Model
Modeling
Business Analyst
Product Owner
Test Team
Designing
Automated Test Design
Generate automation ready test cases And test scripts from model
Test Manager Framework
Automated Test Execution
TR Requirement X
TR Requirement Y
TR Requirement Z
Project Team

FIGURE 4. Automation Test Model

By combining these elements, the ATM significantly enhances the efficiency and effectiveness of the test automation process. It empowers testing teams to focus on higher-value tasks, such as test analysis and strategy, while the ATM takes care of generating and executing test cases with minimal human intervention.

The benefits of the Automation Test Model include:

Consistency and Reusability: By using reusable test assets and automated modeling, the ATM ensures consistency in testing practices across projects. It also promotes the reuse of test cases and resources, saving time and effort in test case creation.

Faster Time-to-Market: With automated test case generation and ready-to-execute test scripts, the ATM accelerates the testing process. This results in faster time-to-market for software releases, enabling organizations to respond quickly to changing market demands.

Reduced Manual Errors: Automation minimizes the risk of manual errors in test case creation and execution, leading to more reliable and accurate testing results.

Adaptability to Changes: The ATM's automated modeling approach allows for easy adaptation to changes in the software application or testing requirements. Test models can be quickly updated and adjusted, ensuring that testing keeps pace with the dynamic nature of software development.

Cost Savings: By automating test case generation and execution, the ATM reduces the need for extensive manual testing efforts, leading to cost savings for organizations.

Overall, the Automation Test Model is a powerful Applied Intelligence model that streamlines test automation and enhances the overall testing process. Its automated modeling and test case generation capabilities provide testing teams with the tools they need to deliver high-quality software efficiently and effectively.

AI-defined test automation brings several advantages, such as increased test coverage, reduced human errors, and better utilization of testing resources. It enables organizations to focus on critical testing activities and strategic decision-making while the intelligent automation takes care of repetitive and time-consuming tasks.

In conclusion, organizations must embrace the concept of AI-defined test automation to achieve higher efficiency, better outcomes, and improved ROI in their testing endeavors. By integrating intelligence into the entire testing process, from scope definition to test case generation and data set management, organizations can unlock the true potential of test automation and deliver high-quality software products in a faster and more cost-effective manner. Now let's see how.

AUTOMATED TEST MODEL CONSTRUCTION

The Automated Test Model Construction is a sophisticated and efficient approach that leverages Artificial Intelligence (AI) to optimize the test case repository and generate a prioritized and dynamic workflow for test execution. This iterative process, driven by the AI model's intelligence, ensures continuous improvement and adaptability in the testing process. Let's break down the key aspects of this approach.

Centralized Test Management Tool (TM): All domain-specific test cases written by Test Engineers or Test Team members are maintained in a central repository within the Test Management tool (TM). This centralized storage ensures better test case organization and accessibility.

Analysis for Redundancy and Optimization: The AI model analyzes the test cases in the TM to identify redundancy and optimization opportunities. By removing duplicate test cases and cataloging similar ones, the model streamlines the test case repository.

Ranking Based on Usage: The AI-driven model assigns a ranking to each test case based on its usage. Frequently used test cases are given higher priority, while less frequently used or redundant test cases may be retired over time.

Dynamic Workflow Generation: Using the optimized test set, the AI model generates a dynamic workflow that determines the set of test cases to be executed based on specific requests. This ensures that the most relevant and essential test cases are executed during each test cycle.

Integration with Code Repository: The workflow generated by the AI model is linked with the code repository. Any changes in the code trigger the model to identify and select the corresponding test cases that need to be modified or executed, ensuring comprehensive test coverage for the code changes.

Iterative Process: The AI model's intelligence allows it to continuously learn and improve over time. It determines which test cases need to be retired based on usage patterns and avoids generating duplicate test sets, promoting a more efficient testing process.

The benefits of this Automated Test Model Construction approach include:

Optimized Test Repository: Redundant and less relevant test cases are removed, leading to a streamlined and well-organized test case repository.

Efficient Test Execution: The prioritized workflow ensures that the most critical test cases are executed first, optimizing test coverage and reducing testing time.

Adaptability to Code Changes: The dynamic workflow ensures that the right test cases are selected for testing code changes, improving regression testing efficiency.

Continuous Improvement: The AI model's iterative nature allows it to adapt to changing testing requirements and improve its performance over time.

Reduced Maintenance Effort: By automatically identifying test cases to be retired and avoiding duplicates, the model reduces the manual effort required for test case maintenance.

In conclusion, the Automated Test Model Construction approach, driven by Artificial Intelligence, enhances test case management, execution, and adaptability in the testing process. Its intelligent and iterative nature ensures continuous improvement and a more efficient testing process, ultimately leading to higher software quality and customer satisfaction.

INTEGRATE AUTOMATED TEST MODEL WITH DEVOPS

Integrating the AI-driven Automation Test Model (ATM) with external API support and DevOps frameworks is a strategic decision that aligns with the principles of agile development. By enabling seamless integration, the ATM can become an asset for any organization seeking to achieve the best return on investment in test automation. Fig.5, as depicted below, showcases how the ATM aims to solve the problem while remaining versatile in its scope beyond network elements.

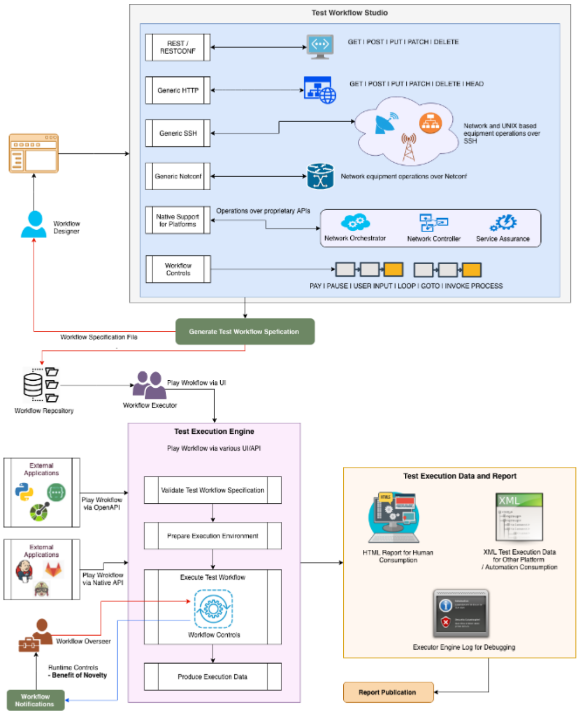

Test Workflow Studio
REST / RESTCONF
GET I POST I PUT I PATCH I DELETE
Generic HTTP
GET I POST I PUT I PATCH I DELETE I HEAD
Generic SSH
Network and UNIX based equipment operations over SSH
Generic Netconf
Network equipment operations over Netconf
Native Support for Platforms
Operations over proprietary APIs
Network Orchestrator
Network Controller
Service Assurance
Workflow Controls
PAY I PAUSE I USER INPUT I LOOP I GOTO I INVOKE PROCESS
Workflow Designer
Generate Test Workflow Specfication
Workflow Specification File
Workflow Repository
Play Wrokflow via UI
Workflow Executor
Test Execution Engine
Play Workflow via various UI/API
External Applications
Play Wrokflow via OpenAPI
External Applications
Play Wroklfow via Native API
Validate Test Workflow Specification
Prepare Execution Environment
Execute Test Workflow
Workflow Controls
Produce Execution Data
Workflow Overseer
Runtime Controls
- Benefit of Novelty
Workflow Notifications
Test Execution Data and Report
HTML Report for Human Consumption
XML
XML Test Execution Data for Other Platform / Automation Consumption
Executor Engine Log for Debugging
Report Publication

FIGURE 5. Understand test workflow engine.

By integrating the ATM with external API support and DevOps frameworks, organizations can unlock the full potential of test automation and achieve seamless collaboration between development and testing teams. This results in faster delivery of high-quality software products, reduced time-to-market, and improved customer satisfaction. The AI-driven Automation Test Model brings efficiency, accuracy, and adaptability to the testing process, making it an asset for any organization's DevOps journey.

SUCCESS STORY

"*Quality is never an accident. It is always the result of intelligent effort*" -- John Ruskin

The Automation Test Model with Applied Intelligence has proven to be highly successful in various internal projects. While the overall concept is still in the proposal stage, a segment covering automated test case generation through a workflow-based model has been implemented and targeted to several software delivery projects. Figure 6 illustrates a snippet of the end-to-end test workflow generated by the Automation Test Model.

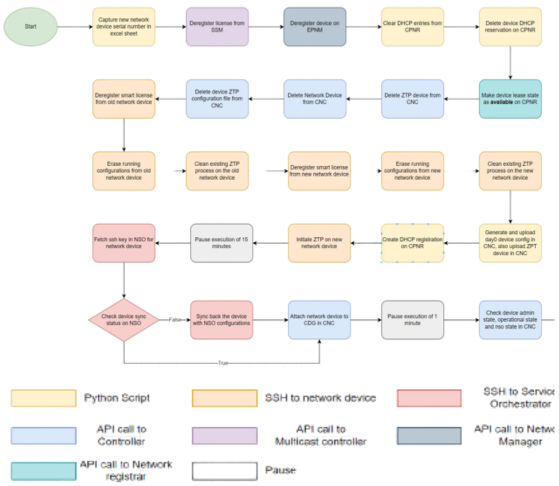
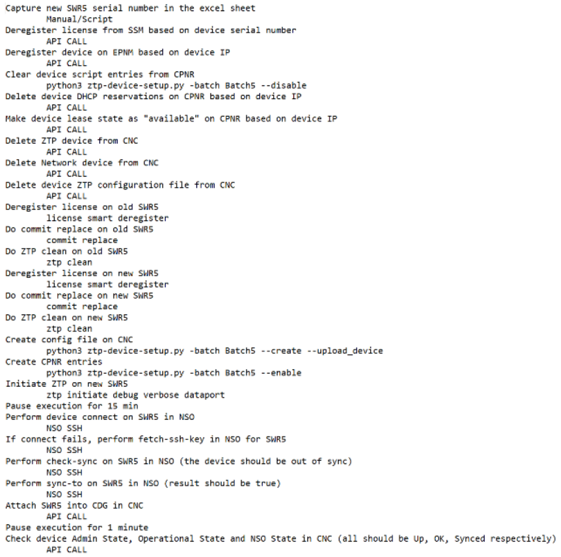

Start
Capture new network device serial number in excel sheet
Deregister license from SSM
Deregister device on EPNM
Clear DHCP entries from CPNR
Delete device DHCP reservation on CPNR
Make device lease state as available on CPNR
Delete ZTP device from CNC
Delete Network Device from CNC
Delete device ZTP configuration file from CNC
Deregister smart license from old network device
Erase running configurations from old network device
Clean existing ZTP process on the old network device
Deregister smart license from new network device
Erase running configurations from new network device
Clean existing ZTP process on the new network device
Generate and upload day0 device config in CNC, also upload ZPT device in CNC
Create DHCP registration on CPNR
Initiate ZTP on new network device
Pause execution of 15 minutes
Fetch ssh key in NSO for network device
Check device sync status on NSO
False
True
Sync back the device with NSO configurations
Attach network device to CDG in CNC
Pause execution of 1 minute
Check device admin state, operational state and nso state in CNC
Python Script
SSH to network device
SSH to Service Orchestrator
API call to Controller
API call to Multicast controller
API call to Network Manager
API call to Network registrar
Pause

FIGURE 6. End-to-end test workflow generated from Automation test model.

The workflow, when translated into human-readable text format, is interpreted as demonstrated in Figure 7. The key advantage is that instead of writing scripts for test execution manually, the workflow is formulated, and the backend engine automatically converts it into an automation script. As a result, there is no need for a dedicated automation test team to generate or modify automated test scripts for execution.

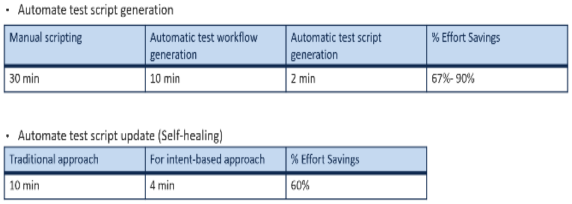
The Automation Test Model's efficiency gains in test case generation, selection, and execution reduce manual efforts and improve testing productivity. This approach has led to substantial savings in time and resources by eliminating the need for a dedicated automation test team. Additionally, the model's AI-driven optimization of test cases and input data sets ensures comprehensive test coverage, contributing to improved software quality and reliability.



```
Capture new SWR5 serial number in the excel sheet
        Manual/Script
Deregister license from SSM based on device serial number
        API CALL
Deregister device on EPNM based on device IP
        API CALL
Clear device script entries from CPNR
        python3 ztp-device-setup.py -batch Batch5 --disable
Delete device DHCP reservations on CPNR based on device IP
        API CALL
Make device lease state as "available" on CPNR based on device IP
        API CALL
Delete ZTP device from CNC
        API CALL
Delete Network device from CNC
        API CALL
Delete device ZTP configuration file from CNC
        API CALL
Deregister license on old SWR5
        license smart deregister
Do commit replace on old SWR5
        commit replace
Do ZTP clean on old SWR5
        ztp clean
Deregister license on new SWR5
        license smart deregister
Do commit replace on new SWR5
        commit replace
Do ZTP clean on new SWR5
        ztp clean
Create config file on CNC
        python3 ztp-device-setup.py -batch Batch5 --create --upload_device
Create CPNR entries
        python3 ztp-device-setup.py -batch Batch5 --enable
Initiate ZTP on new SWR5
        ztp initiate debug verbose dataport
Pause execution for 15 min
Perform device connect on SWR5 in NSO
        NSO SSH
If connect fails, perform fetch-ssh-key in NSO for SWR5
        NSO SSH
Perform check-sync on SWR5 in NSO (the device should be out of sync)
        NSO SSH
Perform sync-to on SWR5 in NSO (result should be true)
        NSO SSH
Attach SWR5 into CDG in CNC
        API CALL
Pause execution for 1 minute
Check device Admin State, Operational State and NSO State in CNC (all should be Up, OK, Synced respectively)
        API CALL
```

FIGURE 7. Workflow generated in readable format.

Based on internal sources of customer deliverables, we can affirm that the utilization of this method has resulted in an overall saving of approximately sixty percent (60%) of effort when compared to conventional test automation techniques. Various use cases have consistently demonstrated positive results, as shown in Figure 8:



- Automate test script generation

Manual scripting	Automatic test workflow generation	Automatic test script generation	% Effort Savings
30 min	10 min	2 min	67%- 90%

- Automate test script update (Self-healing)

Traditional approach	For intent-based approach	% Effort Savings
10 min	4 min	60%

FIGURE 8. Return on Investment

LOOK FORWARD

Developing an intelligent test model is a challenging yet rewarding task for your team. It involves analyzing all existing automation data and creating a predictive test identification model for a specific domain, as depicted in Figure 9. This milestone aims to automatically identify the test cases that need to be executed when code changes or any configuration changes occur.

Milestone 1: Data Analysis and predictive test model creation for a specific domain

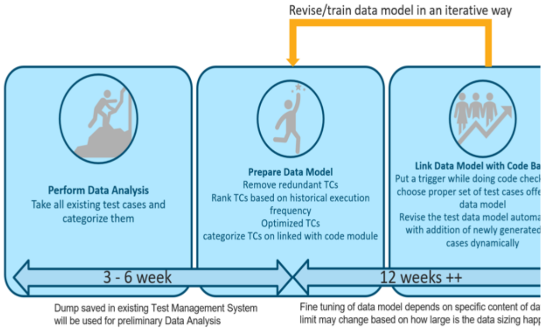

Revise/train data model in an iterative way
Perform Data Analysis
Take all existing test cases and categorize them
Prepare Data Model
Remove redundant TCs
Rank TCs based on historical execution frequency
Optimized TCs
categorize TCs on linked with code module
Link Data Model with Code Ba
Put a trigger while doing code check
choose proper set of test cases offe
data model
Revise the test data model automa
with addition of newly generated
cases dynamically
3 - 6 week
12 weeks ++
Dump saved in existing Test Management System will be used for preliminary Data Analysis
Fine tuning of data model depends on specific content of da
limit may change based on how large is the data sizing happ

FIGURE 9. Predictive Test Model Creation

- The first-year goal is to analyze data from various domains and create an accurate predictive test model. This model will automatically identify the test cases to be executed when code changes or configuration changes happen.
- The model must be dynamic and adaptable, continuously learning from new data to remain up to date.
- Currently, test case identification is manually handled by a subject matter expert. The proposal is to automate this process, eliminating the need for human intervention in selecting test cases.

Milestone 2: Generate workflow based on input generated from selected test cases.

- In the second year, the focus is on generating a workflow based on inputs from the selected test cases. This workflow will automate the test execution process.
- The Automated Test Model (ATM) will be trained with a workflow repository to execute a specific test suite automatically.

Milestone 3: Self-healing test script generation

- Also in the second year, the goal is to achieve self-healing test script generation. The model will automatically detect modifications in the test workflow and generate a self-healing test script.
- The model will be linked with each test selection and an automated test script (job file). It will intelligently fix changes in the job file to align it with the System Under Test (SUT), reducing the need for manual intervention.

The envisioned outcome of this model, as shown in Figure 4, is to automate the entire end-to-end test automation process, not just test execution. With predictive test generation and self-healing test script capabilities, the model aims to deliver a better return on investment across all domains.

By automating test case identification, generating automated workflows, and enabling self-healing test scripts, your team will achieve significant improvements in testing efficiency, accuracy, and cost-effectiveness. This comprehensive approach to test automation will empower your organization to deliver high-quality software with faster release cycles, making it an asset in your software development life cycle.

CONCLUSION

This paper introduces a novel test methodology that emphasizes the concept of "Automate test automation process." Unlike the prevailing industry trend of pursuing automation solely through Artificial Intelligence (AI) and Machine Learning (ML), this approach focuses on effectively aligning automation with the specific business requirements of an organization. The goal is to create a robust and concrete solution that can evolve to leverage NLP (Natural Language Processing) based engines in the future.

The current implementation of this approach has been adopted internally and has already demonstrated significant success. The success story section highlights the positive outcomes achieved through this innovative methodology. Customers, too, are showing a strong demand for faster, more effective, and smarter test solutions as a service.

With this proposal, the paper aims to address the growing customer urge for test solutions that not only meet business targets but also exceed customer expectations in terms of efficiency and quality. By embracing this unique approach, the industry can achieve the desired business outcomes and ensure customer satisfaction, fulfilling the promises made to clients.

The methodology's core pillars include leveraging AI and ML to automate the test automation process, making it smarter, adaptive, and aligned with the organization's specific needs. This approach stands as a beacon of innovation, offering a glimpse into the future where NLP-based engines can further enhance test automation capabilities, elevating the industry's overall testing practices.

The paper demonstrates how the proposed approach can revolutionize the test automation landscape and usher in a new era of streamlined, efficient, and customer-centric testing services. By embracing this methodology, organizations can pave the way for transformative change and achieve remarkable success in their software development endeavors.

ACKNOWLEDGMENTS

We express our sincere gratitude to Mr. Jesse Reed, Mr. Richard Froom, and Ms. Sumitha Prashanth from Cisco Systems Inc. for their invaluable contributions and support in this work. Their expertise and dedication have played a crucial role in the successful development and implementation of the proposed test methodology.

We would also like to acknowledge the support provided by the grant from "Innovation and Unification of Automation Assets of Customer Experience – Digital Assurance and Automation." This grant has been instrumental in facilitating the research and development of the automation-driven test model and has contributed significantly to the positive outcomes achieved.

Their collaboration and assistance have been essential in shaping this paper and bringing the vision of a more efficient and effective test automation process to fruition. We extend our heartfelt appreciation to all the individuals and entities involved in making this project a reality.

AUTHORS DETAILS

Sudipta Debnath is a highly accomplished Technical Leader at Cisco Systems Inc. with extensive expertise in Delivery Automation and Optimization. She has made significant contributions to the field and has published 19 research papers in various reputable forums.

Sudipta's current focus includes spearheading initiatives on Cloud Orchestration, both within her organization and in external collaborations. Her global presence is evident as she is a distinguished speaker at numerous international forums. Sudipta is also an active member of the Women in Engineer group of IEEE, Region 3. For any inquiries, she can be contacted at suddebna@cisco.com.

Debasish Bhadra is a talented Technical Leader at Cisco Systems India Pvt. Ltd., where he has played a pivotal role in driving process automation and optimization throughout his esteemed career in the industry. The concept and idea presented in this paper are his brainchild, showcasing his innovative thinking and problem-solving capabilities. For any communications, Debasish can be reached at debhadra@cisco.com.

Both Sudipta and Debasish's contributions have been instrumental in the development and success of this paper, and their expertise and dedication have been vital in bringing the proposed test methodology to fruition. Their valuable insights and leadership have elevated the automation-driven test model and contributed to its potential as a transformative solution in the field of test automation.

Upcoming Event

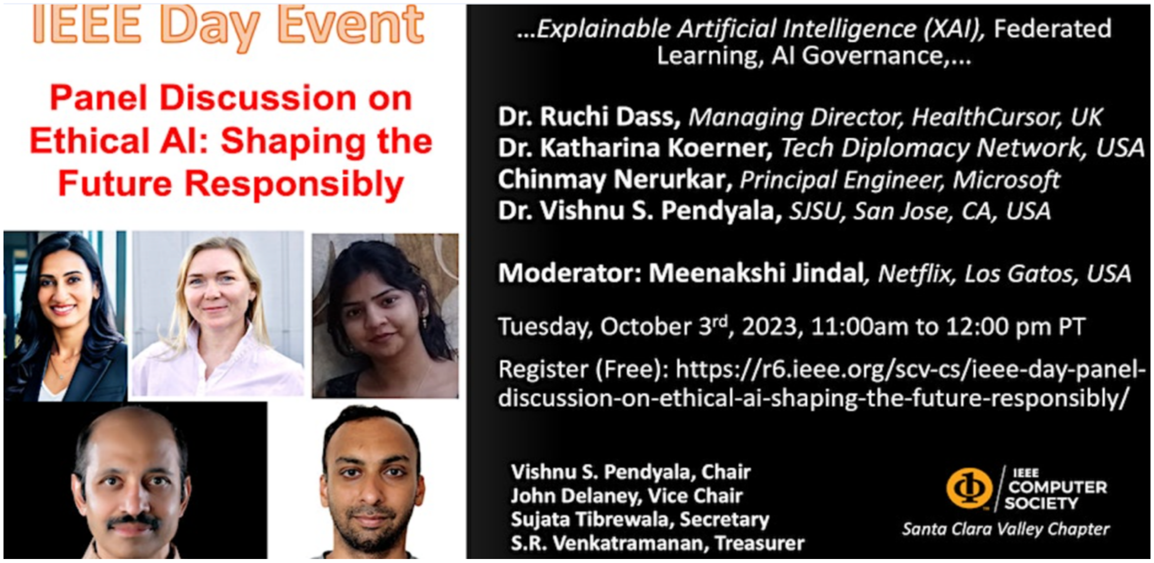
IEEE Day Panel Discussion on Ethical AI: Shaping the Future Responsibly

…Explainable Artificial Intelligence (XAI), Federated Learning, AI Governance, …
With a focus on "Leveraging Artificial Intelligence for a better tomorrow" to align with the theme of the IEEE day.

Dr. Ruchi Dass, *Managing Director, HealthCursor, UK*
Dr. Katharina Koerner, *Tech Diplomacy Network, USA*
Chinmay Nerurkar, *Principal Engineer, Microsoft*
Dr. Vishnu S. Pendyala, *SJSU, San Jose, CA, USA*
Moderator: Meenakshi Jindal, *Netflix, Los Gatos, USA*

Virtual Event via Zoom and YouTube live.
Date: Tuesday, October 3, 2023, 11:00 AM (PT)

Registration: https://www.eventbrite.com/e/ieee-day-panel-discussion-on-ethical-ai-shaping-the-future-responsibly-tickets-682596904717?aff=oddtdtcreator



IEEE Day Event

Panel Discussion on Ethical AI: Shaping the Future Responsibly

…Explainable Artificial Intelligence (XAI), Federated Learning, AI Governance,…

Dr. Ruchi Dass, *Managing Director, HealthCursor, UK*
Dr. Katharina Koerner, *Tech Diplomacy Network, USA*
Chinmay Nerurkar, *Principal Engineer, Microsoft*
Dr. Vishnu S. Pendyala, *SJSU, San Jose, CA, USA*

Moderator: Meenakshi Jindal, *Netflix, Los Gatos, USA*

Tuesday, October 3rd, 2023, 11:00am to 12:00 pm PT

Register (Free): https://r6.ieee.org/scv-cs/ieee-day-panel-discussion-on-ethical-ai-shaping-the-future-responsibly/

Vishnu S. Pendyala, Chair
John Delaney, Vice Chair
Sujata Tibrewala, Secretary
S.R. Venkatramanan, Treasurer

IEEE COMPUTER SOCIETY
Santa Clara Valley Chapter

Building High Performance Modern Web Applications

Ruchi Agarwal, Senior Software Engineer, Netflix Inc, Los Gatos, 95032, USA

Abstract—Building high-performance web applications is vital in today's digital world to meet user expectations and deliver exceptional experiences. This article explores key strategies, architectures, and techniques for achieving high performance. We delve into front-end optimization, covering efficient resource management, image optimization, and browser-based caching. We also discuss approaches to minimize network latency, reduce server load, and modern web techniques for improved performance. By implementing the discussed strategies and best practices, readers will unlock the secrets to deliver lightning-fast, highly responsive web experiences.

INTRODUCTION

In this fast-moving world, there is barely any time for slow stuff, let alone applications and tools on the web that are accessed by billions of people every day. Whether it's a mobile app, an e-commerce platform, or a content-rich web app, users demand high speed and seamless experiences. Slow-loading pages, lagging interactions, and unresponsive interfaces are a surefire way to lose customers and ruin your online reputation. The world of high-performance applications is such where even milliseconds matter a lot. Hence, constructing online applications with exceptional performance is not merely a luxury but a significant requirement. But what does it take to achieve such a feat? How can developers and businesses ensure their web applications are built for speed without sacrificing functionality or user experience? In this article, we will embark on a journey to explore the realm of building high-performance web applications. We will dive deep into the core principles, techniques, and best practices that empower developers to create web applications that load quickly, respond swiftly, and provide users with a delightful experience.

WHAT IT TAKES TO BUILD HIGH PERFORMANCE APPLICATIONS

There are two key factors that cannot be neglected when building world class applications:

Speed: Speed is the number one key to high performance applications. In today’s era speed is almost synonymously used when thinking of performant applications. From conversion rates for eCommerce applications to great user engagement for entertainment or social media-based companies to better ranking on search engines for any sort of business, speed plays a vital role. With a very small attention span of users in today’s world, research shows that a staggering 53% of users will abandon an ongoing session if it takes more than 3 seconds for the application to respond [1]. To meet such an uncompromising and stringent demand, businesses spend thousands to millions of dollars on engineering and design of their applications. It conveys why speed matters so much. It can incur big monetary losses to businesses if speed is not up to the par.

Design: When thinking of performance, design might not be what comes to mind for most people. That is why design gets most neglected when the goal is to just build performant tools and apps.

But why is design so important for an application's performance?

This article stresses a lot on the importance of a well thought and purposeful design because design is extremely crucial in building applications that are efficient, high speed and overall performant. An overly complicated design can not only lead to unintuitive and confusing user experience but also impact the applications performance. A confusing UI can turn away users. This can incur high financial losses for businesses. A well-known example of such a case is Citibank from 2020 where the business lost $500million just because of confusing UI/UX [2].

The more data we let the user interface download to show the user, the slower the response time will be and eventually lets down the user's experience on the application. A well thought and simple design can lead to not only lower latencies in responses but also drives delightful and smooth user experiences. It guides how much data we show or present to the user that the user will find useful.

VARIOUS OPTIMIZATION TECHNIQUES KEEPING SPEED AND DESIGN IN MIND

Now that we have laid out the key contributing factors to an application's performance, we will talk about how to improve these determinants, so our applications are highly efficient and perform at optimization level.

FRONT-END OPTIMIZATION

Front-end or User interface is the gateway to an application. It is through the front-end that a user interacts with an application. Front-end of an application plays a vital role in driving user experience and delivering satisfaction. Hence, optimizing and fine tuning the front-end is not only necessary but also very important in delivering the right experience to users.

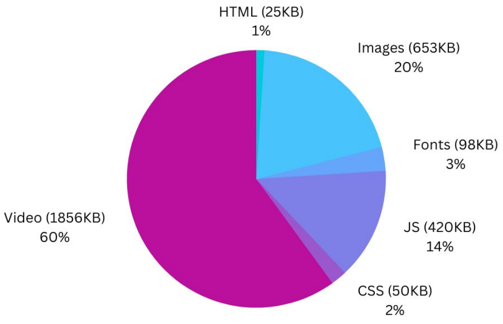
When it comes to front-end optimization, there are several techniques that can be employed to improve application performance. These include:

1. Minimizing HTTP requests: HTTP requests can be very time consuming, reducing the number of results can save us a lot of time in delivering faster page loads. Akamai found that page load time increases directly correlate with lower conversion rates, with every 100 milliseconds increase in load time resulting in a 7% reduction in conversions [3]. Optimizing HTTP requests could thus significantly impact revenue.

   Here are some strategies to reduce HTTP requests:

   a. Caching reusable assets on the client side.
   b. Data URI Scheme enables embedding smaller assets like images into the HTML or CSS itself.
   c. Bundling Javascript and CSS files

2. Optimizing images: Images contribute towards a large portion of the data that is downloaded at the user's end. Images account for over 21% of page load bytes, with the average web page containing over 30 images [4].



Average Bytes per page

   Optimization techniques like compressing images without losing quality, using the right size images for a specific screen size and using the right image format can quickly help with eliminating a lot of slowness incurred by unoptimized images.

   Lossless image compression techniques can reduce image file sizes by up to 40-70%, directly speeding up page load times. Properly sizing images based on screen resolution rather than left at maximum dimensions can yield 40-60% smaller images. (Source: SitePoint).

   Using modern image formats like WebP, which offers 25-34% smaller file sizes than JPEG and 26% smaller than PNG [5].

   By implementing these image optimization techniques, page load times can be improved drastically, and data costs reduced substantially for mobile users, directly impacting user experience and retention.

3. Lazy loading techniques allow non-critical resources such as images and videos to load only when they come into the user's viewport. This means de-prioritizing downloading of images that are below the fold or not close to the user's area of engagement or interaction, improving page load performance. According to Akamai Technologies, Walmart saw up to a 2% increase in conversions for every second of improvement in load time. Every 100ms improvement resulted in up to a 1% increase in revenue [6].

4. To enhance the loading and execution of your JavaScript code, it is essential to optimize it using techniques such as code splitting, tree shaking, and lazy loading.

5. Prioritizing above the fold content of the application: In continuation of the above strategy, prioritizing content that sits above the fold, i.e the content that user will immediately see or interact with which is critical to site's performance. This means, we download and fetch above the fold content first and try to optimize the user's visible assets first. This hugely impacts First Contentful Paint FCP, Time to Interactive TTI and First Meaningful paint FMP.

6. Leveraging browser caching: Using techniques like Local Storage which caches data locally into the browser or caching server sent static assets by instructing the browser which resource to cache and for how long with the usage of certain HTTP headers like Cache-Control, Expires or ETag are some great ways to utilize on premise caching.

7. Minimizing render blocking resources: Render-blocking resources like JavaScript and CSS can significantly delay page rendering. Using strategies such as asynchronous and deferred loading to ensure that these resources don't hinder the initial paint and interactivity of your web application can be hugely impactful in time to first load the initial page for a user.

The above are some of the great approaches through which we can improve efficiency in terms of response and load time of the Front-end. Other than engineering optimization techniques, UI/UX design improvements can significantly contribute towards an application's performance.

Let's look at which design elements can drive such impact:

1. Intuitive design: The key to high-performance applications lies in adopting a simple and clean design. Such design not only reduces cognitive load for users while navigating the applications but also enhances user engagement. A well-executed simplistic design not only facilitates easy improvements but also reduces clutter by presenting the right amount of data to the user. This significantly aids in downloading only what is necessary at the client's end.

2. Consistency: Design Patterns whether it is fonts, icons or UI element styles, it is important to keep consistency throughout the app so there is minimal cognitive friction for the user to understand how to interact and navigate the application. This also helps users to find the information they are looking for and businesses to drive conversions quickly.

3. Responsiveness: A responsive User interface is not only a nice to have feature for application, but in today's world it is a necessity. A study shows in 2019, almost 53% of the total web traffic came from mobile users [7]. This shows that optimizing the UI for only desktop users is not enough when more than half of the user base is mobile based which means unoptimized mobile experience can lead to loss in users. Other than that, an adaptive design can drive how engineering should focus on optimizing assets that are delivered to different device-based users. Reducing and fine tuning the amount of content and assets downloadable at the client side is of utmost importance when thinking about these metrics.

By employing these techniques, developers can significantly reduce page load times and enhance the overall performance of web applications.

BACK-END OPTIMIZATION

Building high performance web applications goes beyond just front-end optimizations. An optimized backend is like a powerhouse that supplies electricity to all components of an application. Hence it is very crucial that the powerhouse is highly optimized, calibrated to meet the needs of the other components.

The following is a list of backend optimization techniques that can be leveraged to deliver a seamless and delightful user experience and create applications that stand the test of time.

1. Load Balancing: To keep the application scalable with demand, it is important that proper load balancing strategies are put in place. This not only avoids unnecessary choking and bottlenecks but also helps with efficiency in request management so that server can respond to all requests in time and without latency incurred by waiting for resources, which in turn improves the performance and availability of applications, databases, and other computing resources [8]

2. Database Optimization: There are several techniques to optimize databases. Based on the requirements and needs of an application and requests, appropriate techniques must be employed.

    a. Indexing to improve read operations. Appropriately indexing the database tables in cases of OLTP relational databases, helps with faster read transactions on the table and find information relatively quickly. When we create a primary key in a table, a clustered index tree is created and all data pages containing the table rows are physically sorted in the file system according to their primary key values [9]. Separate indexes on non-primary keys can also be created based on need, but over indexing can lead to slower writes.

    b. Replication and Redundancy of data to ensure that the data is always available and accessible to users from multiple servers, hence mitigating the chances of a single point of failure. The biggest benefit of database replication is that queries on the database can be processed in parallel by replicas. And because the load is distributed, it helps mitigate load on one server and helps requests being served faster [10]. The servers communicate among each other to achieve consistency through various ways. A popular way to replicate is primary-secondary where write transaction requests can be handled by the primary and read transactions can be handled by secondary or replicas.

    c. Denormalization of data to improve read times. Join queries can be expensive and time consuming which adds read latency. Denormalization can be done by storing derived or redundant data along with the original data to avoid expensive joins or complex data transformations later.

    d. Caching: Optimize database queries by caching the results of frequently executed read queries can significantly reduce the load on the database.

3. CDNs: The power of CDNs cannot be undermined. Utilizing CDNs can be extremely effective in improving web performance, especially because of their ability to store static assets like images, CSS, JS etc and deliver them to end users through a geographically distributed network of servers.

4. Server-Side Solutions: Consider server-side optimizations such as HTTP/2, which allows multiple requests to be multiplexed over a single connection, minimizing the overhead associated with establishing new connections. Additionally, explore server-level compression techniques like gzip to further reduce the size of transferred assets.

5. Messaging Queues: Use messaging queues like RabbitMQ or Kafka to offload time-consuming and resource-intensive tasks from the main web application and database. This ensures better scalability and enhances overall user experience.

CONCLUSION

In conclusion, building high performance modern web applications requires a combination of factors. The key contributors are speed and design. To improve the speed of web applications, front-end optimization techniques such as minimizing HTTP requests, optimizing images, lazy loading, and leveraging browser caching can be employed. Additionally, UI/UX design improvements such as intuitive design, consistency, and responsiveness can contribute significantly towards an application's performance.

For backend optimization, load balancing, database optimization, server-side solutions, and messaging queues can be leveraged to deliver a seamless and delightful user experience.

Employing these techniques can help developers create web applications that load quickly, respond swiftly, and provide users with a delightful experience.

REFERENCES:

[1] Castro, Jay. 2016. "Increase the speed of your mobile site with this toolkit." Increase the speed of your mobile site with this toolkit. https://blog.google/products/ads-commerce/increase-speed-of-your-mobile-site-wi/.

[2] Lee, Timothy B. 2021. "Citibank just got a $500 million lesson in the importance of UI design." Ars Technica. https://arstechnica.com/tech-policy/2021/02/citibank-just-got-a-500-million-lesson-in-the-importance-of-ui-design/.

[3] "Akamai Online Retail Performance Report." 2017. Akamai. https://www.akamai.com/newsroom/press-release/akamai-releases-spring-2017-state-of-online-retail-performance-report.

[4]"HTTP Archive: Page Weight." n.d. The HTTP Archive. Accessed July 17, 2023. https://httparchive.org/reports/page-weight#reqImg.
[5] "WebP files explained | Google's web image format." n.d. Adobe. Accessed July 17, 2023. https://www.adobe.com/creativecloud/file-types/image/raster/webp-file.html.

[6] "The State of Online Retail Performance | Spring 2017 | Akamai.",. Accessed 31 July 2023.

[7] Bouchrika, Imed. 2023. "Mobile vs Desktop Usage Statistics for 2023." Mobile vs Desktop Usage Statistics for 2023. https://research.com/software/mobile-vs-desktop-usage.

[8] "Load Balancing." n.d. IBM. Accessed July 17, 2023. https://www.ibm.com/topics/load-balancing.

[9] Rahman, Syedur, A. M. A. Feroz, Md Kamruzzaman3, and Meherun N. Faruque 4. 2010. "Analyze Database Optimization Techniques." *IJCSNS International Journal of Computer Science and Network Security* 10, no. 8 (August): 1,2.

[10] "Data Replication and its Impact on Business Strategy." n.d. Stitch Data. Accessed July 17, 2023. https://www.stitchdata.com/resources/data-replication/.

ABOUT THE AUTHOR:

Ruchi Agarwal is a Senior Software Engineer at Netflix. With over 11 years of experience in the tech industry, Ruchi has been a pinnacle in the development of many software applications and tools across many high-tech companies including Netflix, Apple, eBay, and TCS. At Netflix, Ruchi works on building Enterprise Applications to help the business Streamline Internal Operations and Work

Acknowledgment of Reviewers

We would like to express our sincere appreciation to the following reviewers who generously dedicated their time and expertise to review the featured papers. Their commitment to scholarly excellence, dedication to the peer-review process, and valuable insights have significantly contributed to the high quality of the articles presented herein:

- **Raghavan Muthuregunathan - Senior Engineering Manager, Search AI**
- **Meenakshi Jindal - Senior Software Engineer, Netflix**
- **Anuj Phadke- Senior Software Engineer, Netflix**

Upcoming Event

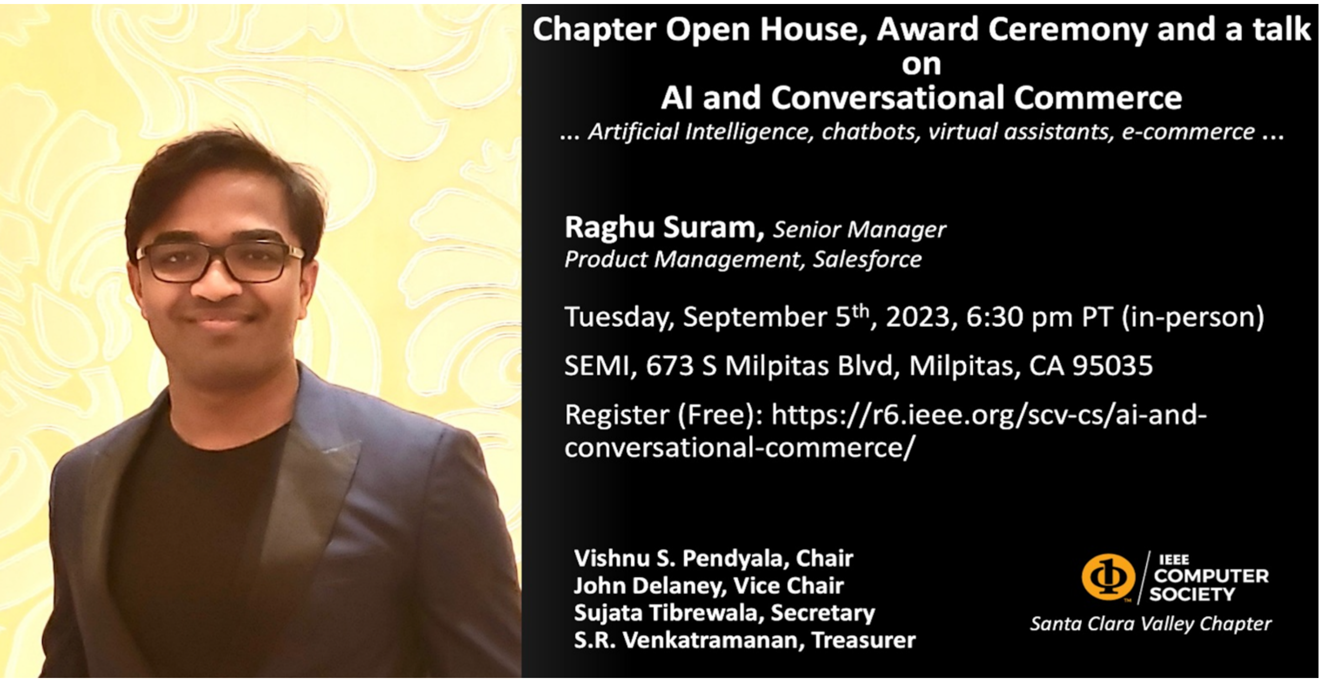
Chapter Open house, Awards Ceremony, and talk on AI and Conversational Commerce

… *Artificial Intelligence, chatbots, virtual assistants, e-commerce…*

Speaker: **Raghu Suram,** *Senior Manager*
Product Management, Salesforce

Date: Tuesday, September 5, 2023, 6:30 PM (PT)

Free Registration: https://www.eventbrite.com/e/ai-and-conversational-commerce-tickets-694491210907



Chapter Open House, Award Ceremony and a talk on AI and Conversational Commerce

… Artificial Intelligence, chatbots, virtual assistants, e-commerce …

Raghu Suram, *Senior Manager*
Product Management, Salesforce

Tuesday, September 5th, 2023, 6:30 pm PT (in-person)

SEMI, 673 S Milpitas Blvd, Milpitas, CA 95035

Register (Free): https://r6.ieee.org/scv-cs/ai-and-conversational-commerce/

Vishnu S. Pendyala, Chair
John Delaney, Vice Chair
Sujata Tibrewala, Secretary
S.R. Venkatramanan, Treasurer

IEEE COMPUTER SOCIETY
Santa Clara Valley Chapter

IEEE COMPUTER SOCIETY

Santa Clara Valley Chapter

Slowly changing dimensions and fast changing facts - the story of the traditional data warehouse

... Data warehouse, Data mart, OLAP, Star schema, Data Analytics ...

Dr. Vishnu S. Pendyala, *San Jose State University, CA, USA*
IEEE Computer Society Distinguished Contributor

Tuesday, August 15th, 2023, 6:00 pm PT

Register (Free): https://r6.ieee.org/scv-cs/slowly-changing-dimensions-and-fast-changing-facts-the-story-of-the-traditional-data-warehouse/

Chair

Vishnu S Pendyala, PhD

Vice Chair

John Delaney

Secretary

Sujata Tibrewala

Treasurer

SR Venkatramanan

Webmaster

Paul Wesling

Connect with us

https://r6.ieee.org/scv-cs/

https://www.linkedin.com/groups/2606895/

http://listserv.ieee.org/cgi-bin/wa?SUBED1=cs-chap-scv&A=1

http://www.youtube.com/user/ieeeCSStaClaraValley

https://www.linkedin.com/company/ieee-computer-society-scv-chapter/