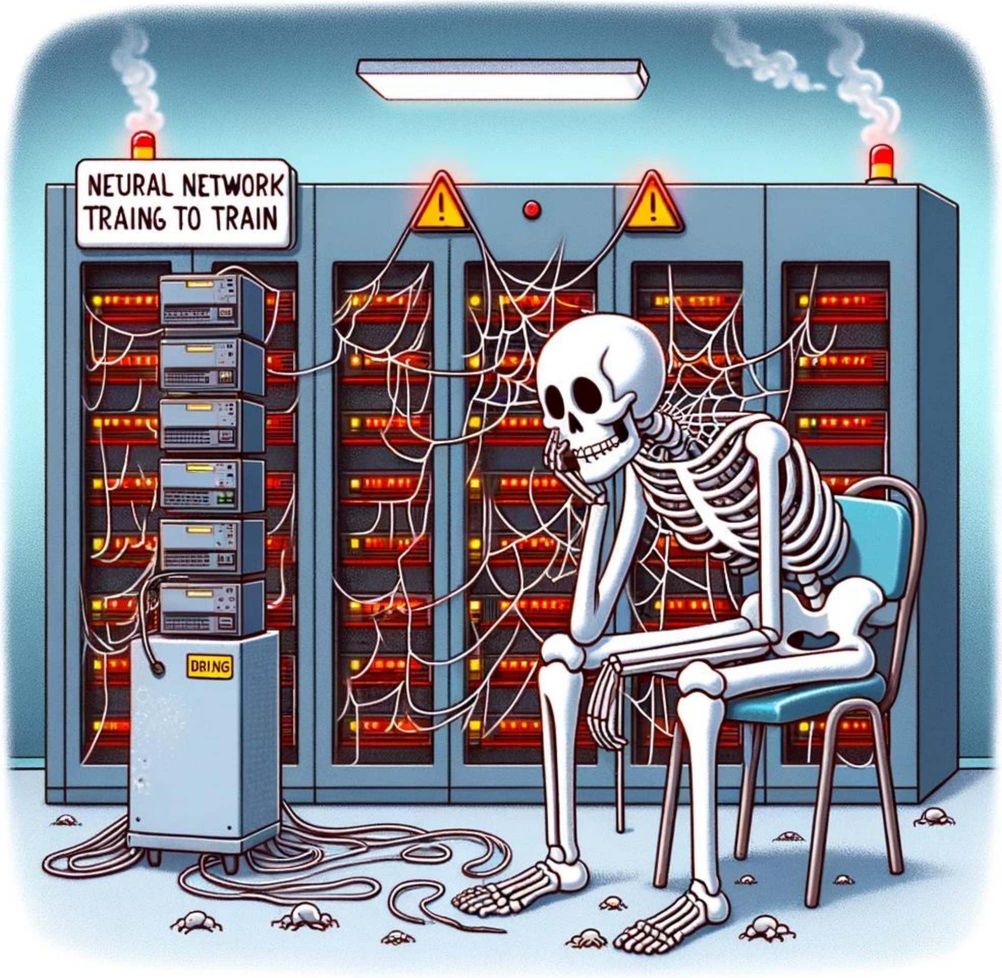
Schrödinger's FP: Dynamic Adaptation of Floating-Point Containers for Deep Learning Training

Miloš Nikolić, Enrique Torres Sanchez, Jiahui Wang, Ali Hadi Zadeh,
Mostafa Mahmoud, Ameer Abdelhadi, Kareem Ibrahim, and Andreas Moshovos

The Edward S. Rogers Sr. Department
of Electrical & Computer Engineering
UNIVERSITY OF TORONTO

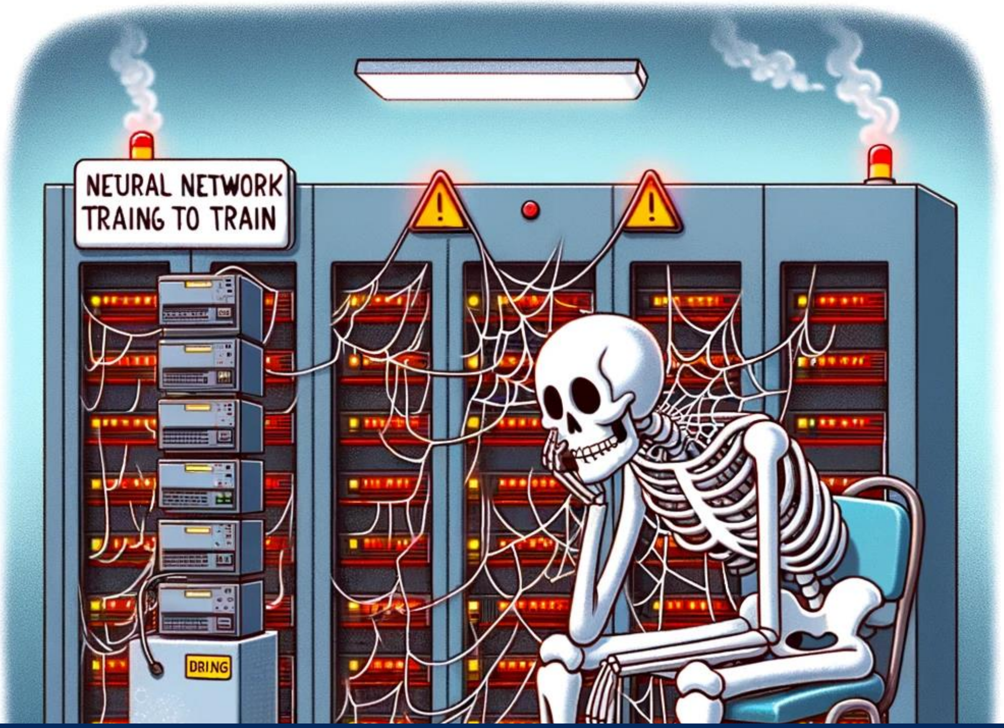
Training is Expensive!

- Neural Networks are becoming ubiquitous
- But they are expensive
  - Energy
  - Time



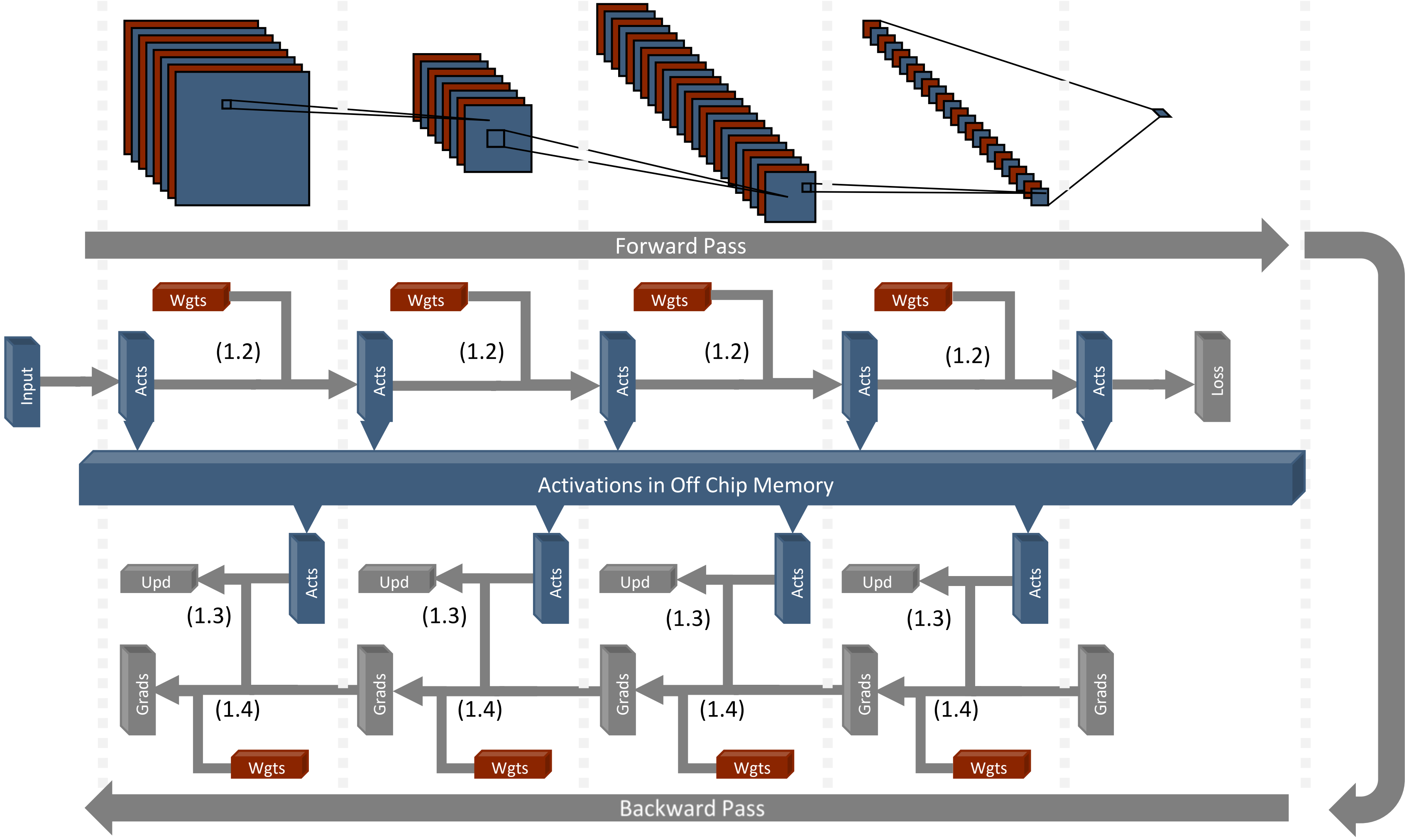
Training is Expensive!

- Neural Networks are becoming ubiquitous
- But they are expensive
  - Energy
  - Time



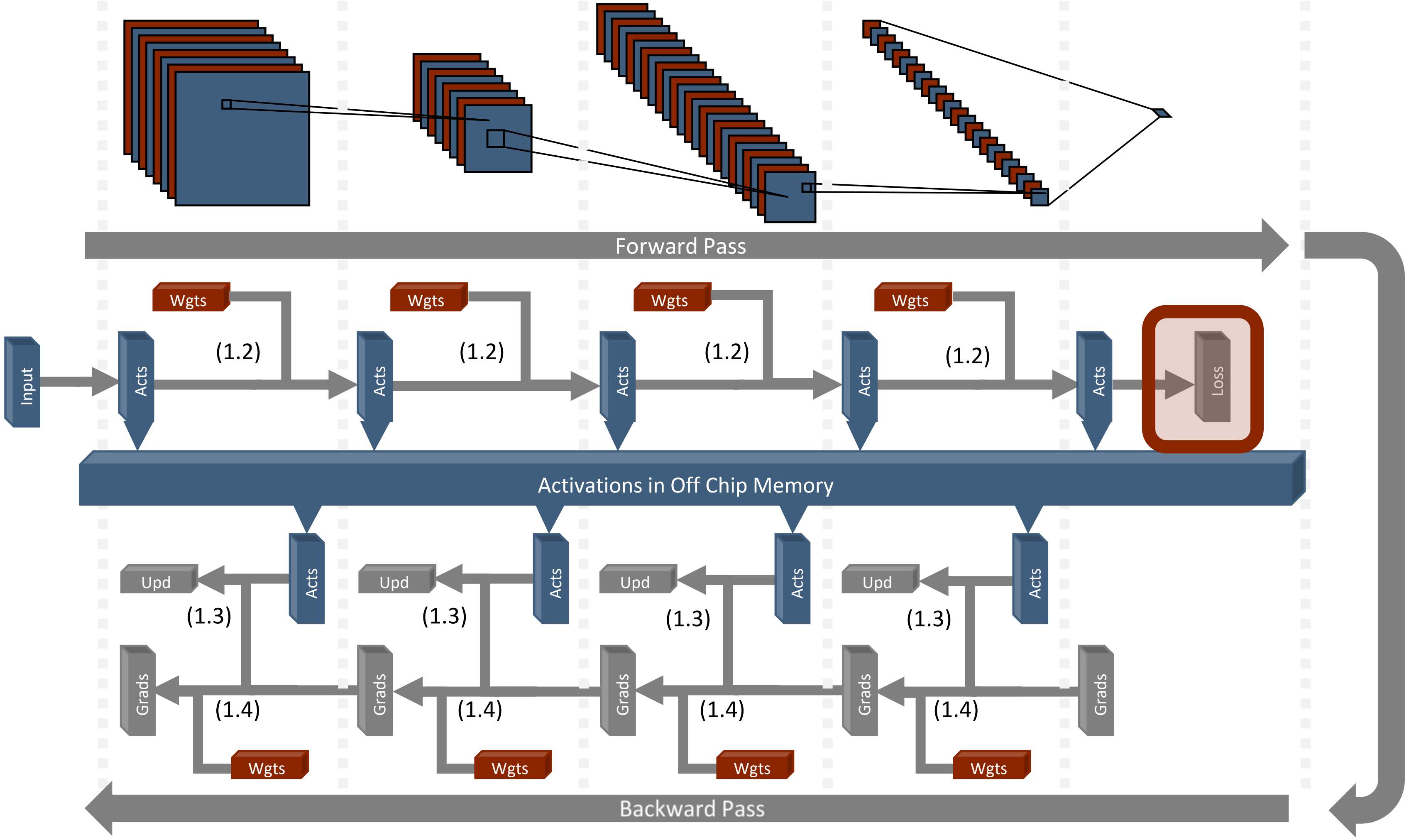
Goal: Improve energy efficiency and performance of Training

Gradient Descent – Overview



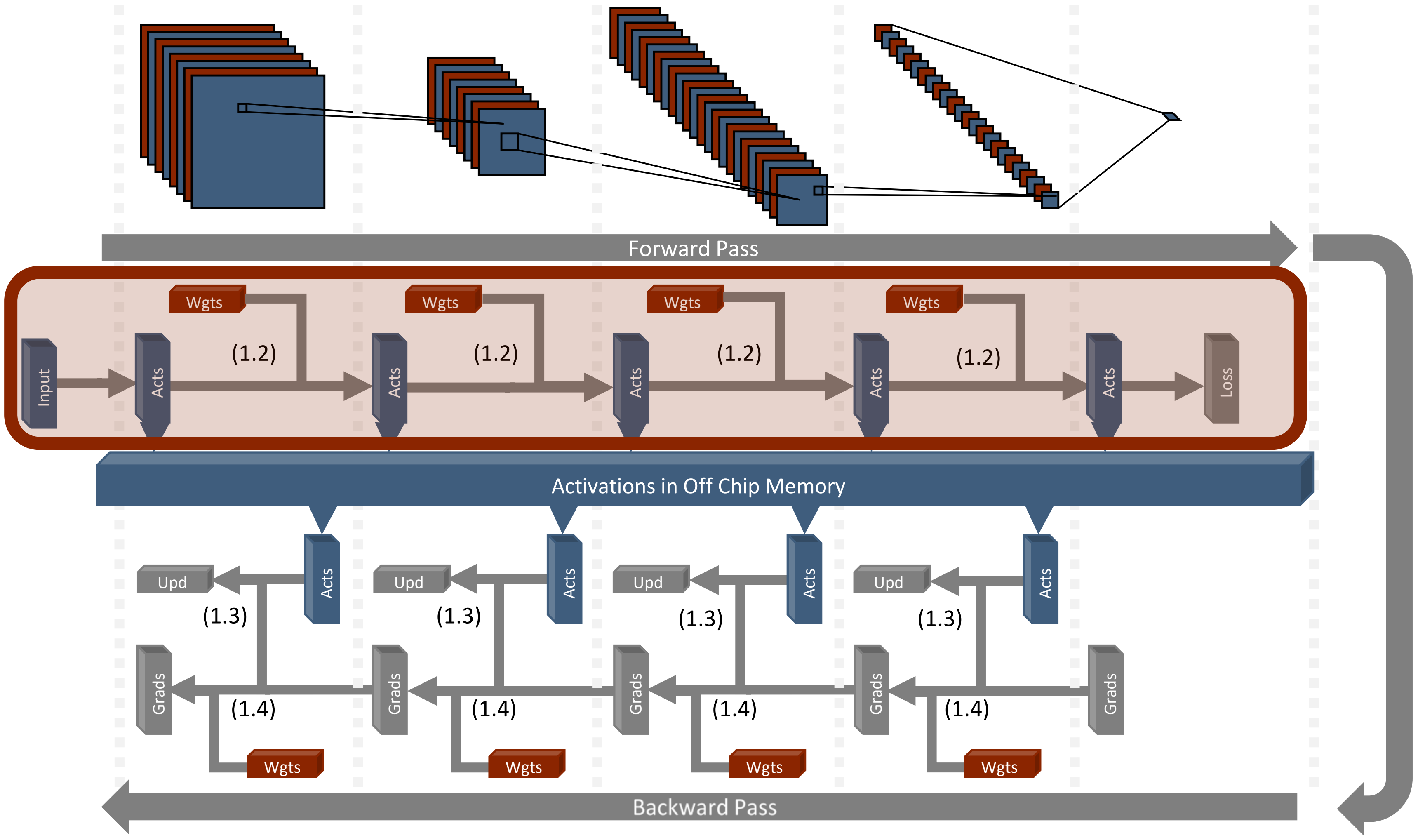
Gradient Descent – Overview

- Loss function



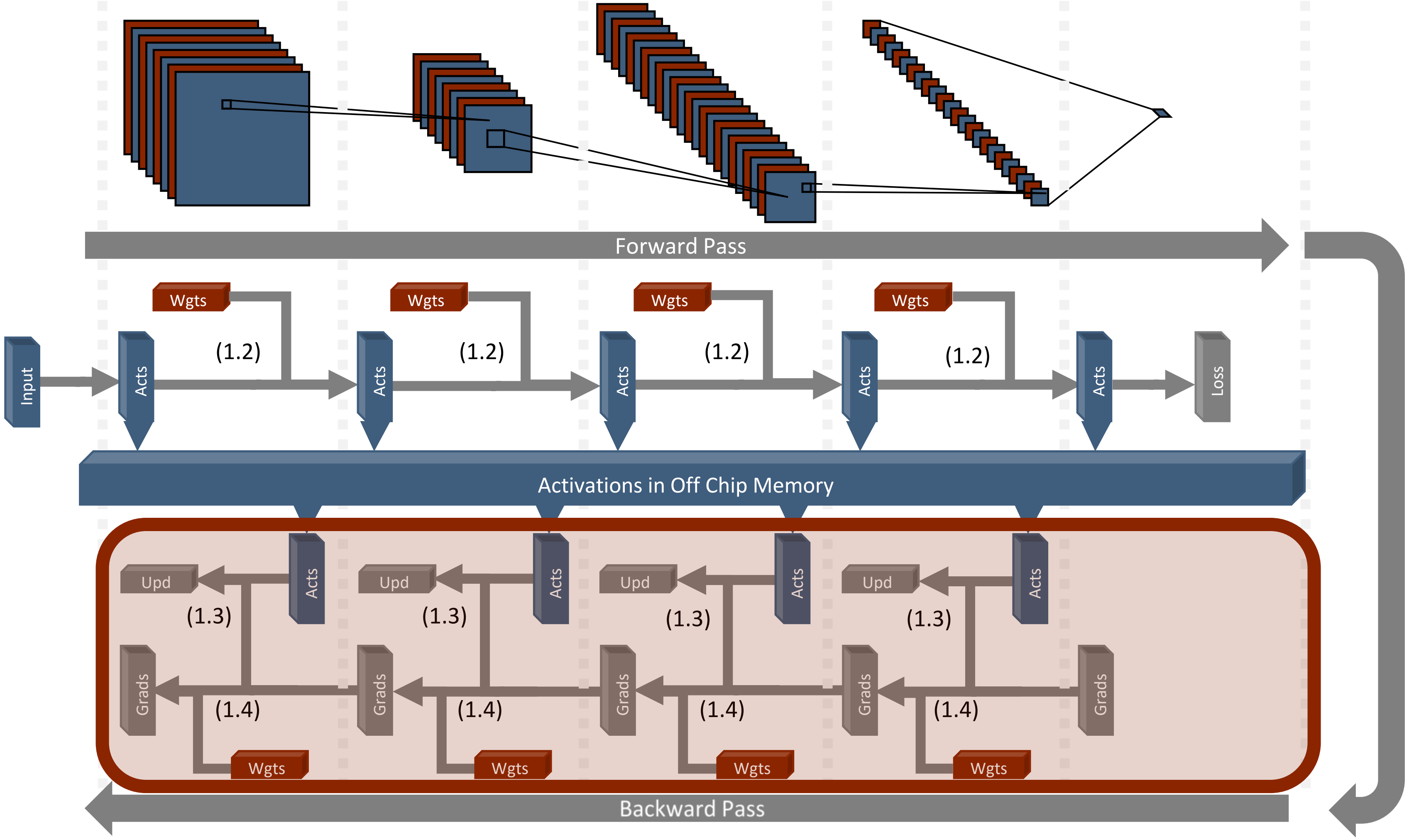
Gradient Descent – Overview

- Loss function
- Forward pass



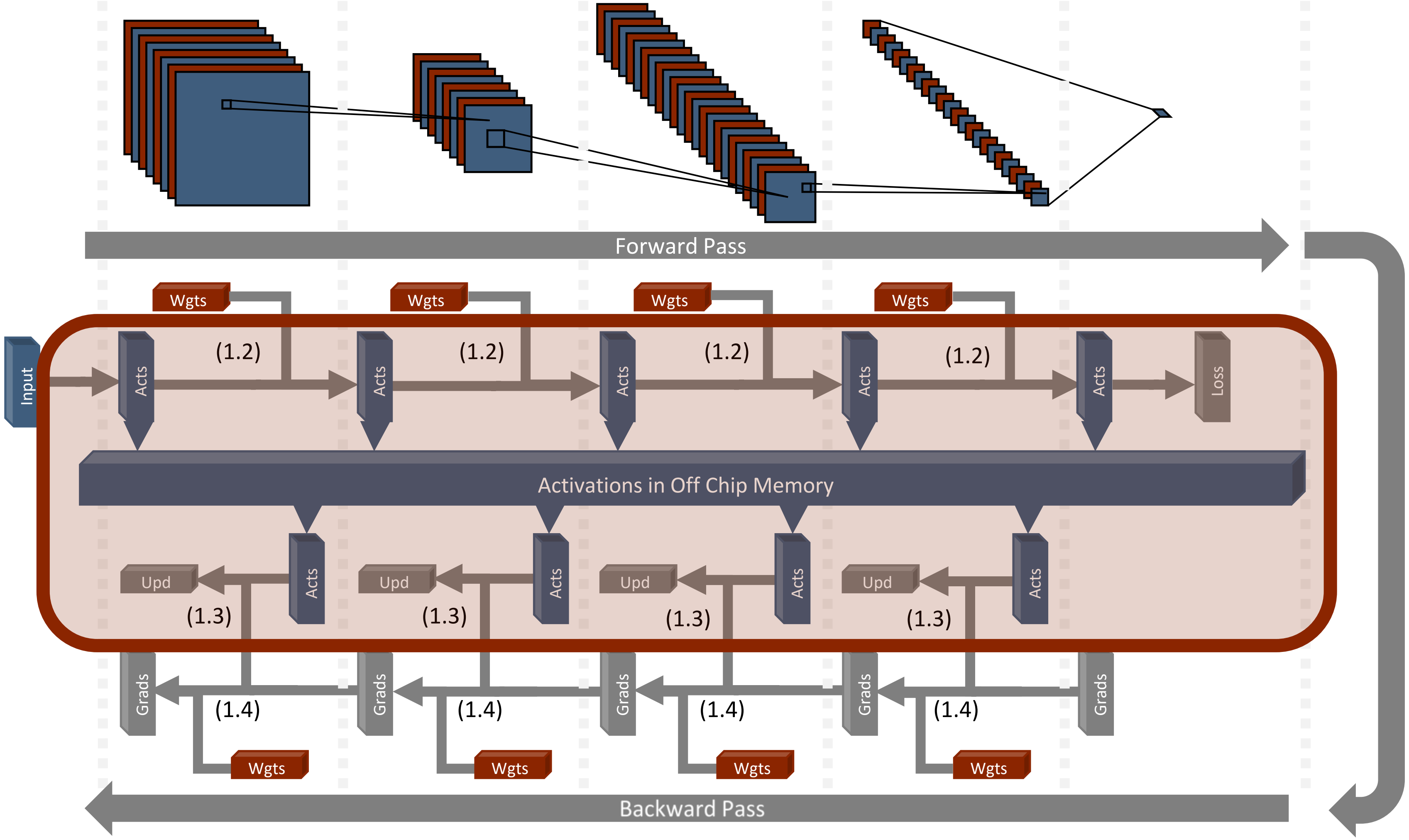
Gradient Descent – Overview

- Loss function
- Forward pass
- Backward pass
    - Update
    - Gradient



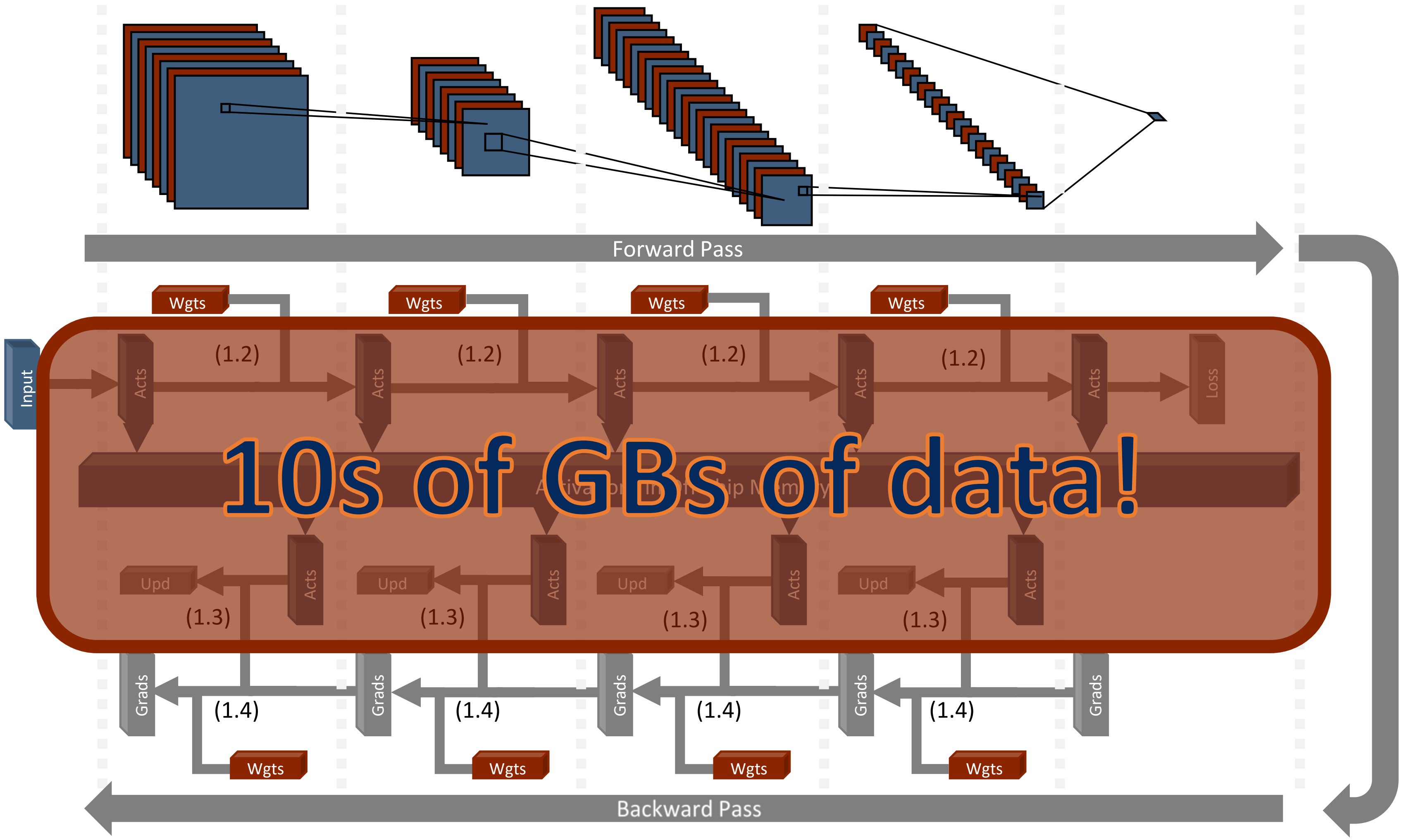
Gradient Descent – Overview

- Loss function
- Forward pass
- Backward pass
  - Update
  - Gradient



Gradient Descent – Overview

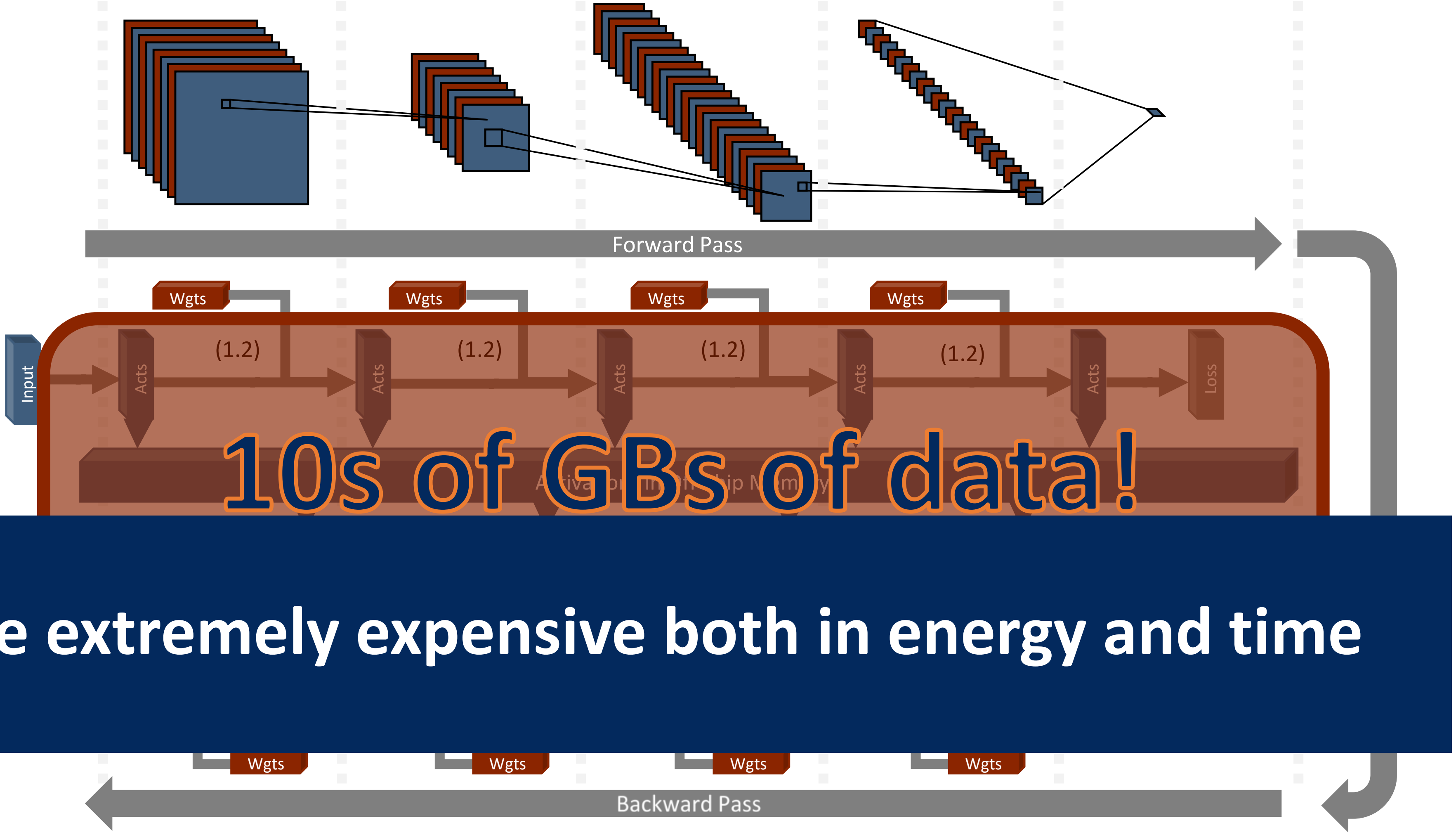
- Loss function
- Forward pass
- Backward pass
  - Update
  - Gradient



10s of GBs of data!

Gradient Descent – Overview

- Loss function
- Forward pass
- Backward pass
  - Update
  - Gradient



10s of GBs of data!

Off-chip accesses are extremely expensive both in energy and time

Narrow Datatypes can help

- Efficient Datatype reduce costs!
  - Memory
  - Compute

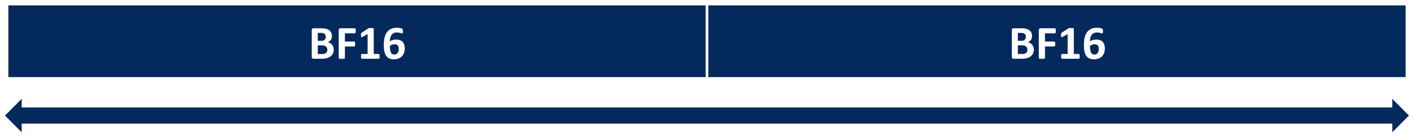
Narrow Datatypes can help

- Efficient Datatype reduce costs!
  - Memory
  - Compute

FP32

Narrow Datatypes can help

- Efficient Datatype reduce costs!
  - Memory
  - Compute

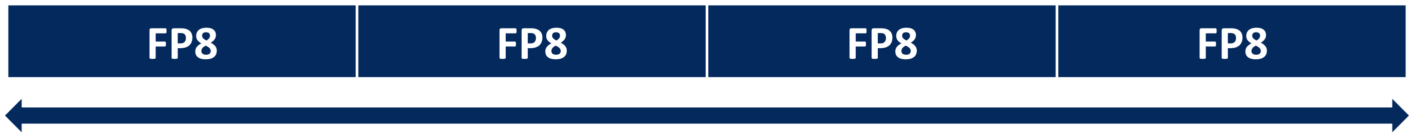


BF16

BF16

Narrow Datatypes can help

- Efficient Datatype reduce costs!
  - Memory
  - Compute



FP8	FP8	FP8	FP8

Narrow Datatypes can help

- Efficient Datatype reduce costs!
    - Memory
    - Compute

FP6 | FP6 | FP6 | FP6 | FP6 | …

Which Datatype? Where? When?

- There are many great options

Published as a conference paper at ICLR 2018

MIXED PRECISION TRAINING

Sharan Narang*, Gregory Diamos, Erich Elsen†
Baidu Research
{sharan, gdiamos}@baidu.com

Paulius Micikevicius*, Jonah Alben, David Garcia, Boris Ginsburg, Michael Houston, Oleksii Kuchaiev, Ganesh Venkatesh, Hao Wu
NVIDIA
{pauliusm, alben, dagarcia, bginsburg, mhouston, okuchaiev, gavenkatesh, skyw}@nvidia.com

ABSTRACT

Training DNNs with Hybrid Block Floating Point

Mario Drumond
Ecocloud
EPFL
mario.drumond@epfl.ch

Tao Lin
Ecocloud
EPFL
tao.lin@epfl.ch

Martin Jaggi
Ecocloud
EPFL
martin.jaggi@epfl.ch

Babak Falsafi
Ecocloud
EPFL
babak.falsafi@epfl.ch

Abstract

The wide adoption of DNNs has given birth to … forcing datacenter operators to adopt do… These accelerators typically emplo… arithmetic to maximize perf… further increase that per… point arithmetic. H… point's narro… We iden…

Training Deep Neural Networks with 8-bit Floating Point Numbers

Naigang Wang, Jungwook Choi, Daniel Brand, Chia-Yu Chen and Kailash Gopalakrishnan
IBM T. J. Watson Research Center
Yorktown Heights, NY 10598, USA
{nwang, choij, danbrand, cchen, kailash}@us.ibm.com

Abstract

The state-of-the-art hardware platforms for training Deep Neural Networks (DN… are moving from traditional single precision (32-bit) computati… bits of precision – in large part due to the high energ… bit storage associated with using reduced-pr… unlike inference, training with numbe… challenging due to the need… back-propagation… of DNN…

A Study of BFLOAT16 for Deep Learning Training

Dhiraj Kalamkar[1], Dheevatsa Mudigere[2], Naveen Mellempudi*[1], Dipankar Das[1], Kunal Banerjee[1], Sasikanth Avancha[1], Dharma Teja Vooturi†[1], Nataraj Jammalamadaka†[1], Jianyu Huang[2], Hector Yuen[2], Jiyan Yang[2], Jongsoo Park[2], Alexander Heinecke[1], Evangelos Georganas[1], Sudarshan Srinivasan[1], Abhisek Kundu[1], Misha Smelyanskiy[2], Bharat Kaul[1], and Pradeep Dubey[1]

[1]Parallel Computing Lab, Intel Labs
[2]Facebook, 1 Hacker Way, Menlo Park, CA

Abstract

This paper presents the first comprehensive empirical st… efficacy of the Brain Floating Point (BF1… Learning training…

Ultra-Low Precision 4-bit Training of Deep Neural Networks

Xiao Sun, Naigang Wang, Chia-yu Chen, Jia-min Ni, Ankur Agrawal, Xiaodong Cui, Swagath Venkataramani, Kaoutar El Maghraoui, Vijayalakshmi Srinivasan, Kailash Gopalakrishnan
IBM T. J. Watson Research Center, Yorktown Heights, NY 10598, USA
{xsun, nwang, cchen, jiamin.ni, ankuragr, cuix, swagath.venkataramani, kelmaghr, viji,kailash}@us.ibm.com

Abstract

…of novel techniques and numerical repres… time, the precision of training systems … To enable this advance, … (GradScale) that addres… …zed gradients as … model trainin…

The 28th IEEE Int. Symp. on High-Performance Computer Architecture (HPCA-28), February 2022

FAST: DNN Training Under Variable Precision Block Floating Point with Stochastic Rounding

Sai Qian Zhang[1], Bradley McDanel[2], and H.T. Kung[1]
[1]Harvard University
[2]Franklin and Marshall College

Abstract—Block Floating Point (BFP) can efficiently support quantization for Deep Neural Network (DNN) training by providing a wide dynamic range via a shared exponent across a group of values. In this paper, we propose a Fast First, Accurate Second Training (FAST) system for DNNs, where the weights, activations, and gradients are represented in BFP. FAST supports matrix multiplication with variable precision BFP input operands, enabling incremental increases in DNN precision throughout training. By increasing the BFP precision across both training iterations and DNN layers, FAST can greatly shorten the training time while reducing overall hardware resource usage. Our FAST Multiplier-Accumulator (fMAC) supports dot product computations under multiple BFP precisions. We validate our FAST system on multiple DNNs with different datasets, demonstrating a 2-6× speedup in training on a single-chip platform over prior work based on mixed-precision or block floating point number systems while achieving similar performance in validation accuracy.

Fig. 1. (left) Overview of the proposed FAST system for DNN … variable precision BFP with stochastic rounding. A group of … converted to a BFP group with a selected mantissa width base… error tolerance (see Algorithm 1). (right) FAST incrementall… BFP precision across both layer depths and training iteratio… on corresponding empirical data obtained via DNN trainin…

FP8 FORMATS FOR DEEP LEARNING

Paulius Micikevicius, Dusan Stosic, Patrick Judd, John Kamalu, Stuart Oberman, Mohammad Shoeybi, Michael Siu, Hao Wu
NVIDIA
{pauliusm, dstosic, pjudd, jkamalu, soberman, mshoeybi, msiu, skyw}@nvidia.com

Neil Burgess, Sangwon Ha, Richard Grisenthwaite
Arm
{neil.burgess, sangwon.ha, richard.grisenthwaite}@arm.com

Naveen Mellempudi, Marius Cornea, Alexander Heinecke, Pradeep Dubey
Intel
{naveen.k.mellempudi, marius.cornea, alexander.heinecke, pradeep.dubey}@intel.com

ABSTRACT

FP8 is a natural progression for accelerating deep learning training inferen… formats common in modern processors. In this paper we propose an 8-bi… interchange format consisting of two encodings - E4M3 (4-bi… (5-bit exponent and 2-bit mantissa). While E5M2 f… special values, E4M3's dynamic range i… mantissa bit-pattern for NaNs… and language tasks… study cov…

…9.05433v2 [cs.LG] 29 Sep 2022

With Shared Microexponents, A Little Shifting Goes a Long Way

…uhani*, Ritchie Zhao, Venmugil Elango, Rasoul Shafipour, Mathew Hall, Maral Mesmakhosroshahi, …More, Levi Melnick, Maximilian Golub, Girish Varatkar, Lei Shao, Gaurav Kolhe, Dimitry Melts, Jasmine Klar, Renee L'Heureux, Matt Perry, Doug Burger, Eric Chung
Microsoft
Zhaoxia (Summer) Deng, Sam Naghshineh, Jongsoo Park, Maxim Naumov
Meta

…stract—This paper introduces Block Data Representations …R), a framework for exploring and evaluating a wide spec… …n of narrow-precision formats for deep learning. It enables …parison of popular quantization standards, and … …R, new formats based on …

Many data formats have been proposed, and the … space of possible new idea…

Which Datatype? Where? When?

- There are many great options
- But it’s a trade-off
  - Accuracy vs. Efficiency
  - All have edge cases

Mixed Precision Training

Training DNNs with Hybrid Block Floating Point

Training Deep Neural Networks with 8-bit Floating Point Numbers

A Study of BFLOAT16 for Deep Learning Training

Ultra-Low Precision 4-bit Training of Deep Neural Networks

FAST: DNN Training Under Variable Precision Block Floating Point with Stochastic Rounding

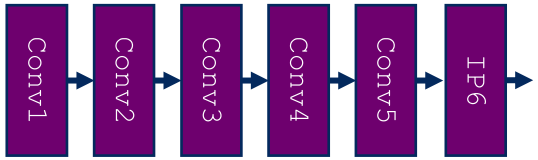
FP8 Formats For Deep Learning

With Shared Microexponents, A Little Shifting Goes a Long Way

Which Datatype? Where? When?

- There are many great options
- But it's a trade-off
  - Accuracy vs. Efficiency
  - All have edge cases

FP32						
BF16	✓					✓
FP8		✓		✓		
FP6			✓		✓	



Published as a conference paper at ICLR 2018

MIXED PRECISION TRAINING

Sharan Narang*, Gregory Diamos, Erich Elsen†
Baidu Research
{sharan, gdiamos}@baidu.com

Paulius Micikevicius*, Jonah Alben, David Garcia, Boris Ginsburg, Michael Houston, Oleksii Kuchaiev, Ganesh Venkatesh, Hao Wu
NVIDIA
{pauliusm, alben, dagarcia, bginsburg, mhouston, okuchaiev, gavenkatesh, skyw}@nvidia.com

ABSTRACT

Training DNNs with Hybrid Block Floating Point

Mario Drumond, Ecocloud, EPFL, mario.drumond@epfl.ch
Tao Lin, Ecocloud, EPFL, tao.lin@epfl.ch
Martin Jaggi, Ecocloud, EPFL, martin.jaggi@epfl.ch
Babak Falsafi, Ecocloud, EPFL, babak.falsafi@epfl.ch

Abstract

Training Deep Neural Networks with 8-bit Floating Point Numbers

Naigang Wang, Jungwook Choi, Daniel Brand, Chia-Yu Chen and Kailash Gopalakrishnan
IBM T. J. Watson Research Center
Yorktown Heights, NY 10598, USA
{nwang, choij, danbrand, cchen, kailash}@us.ibm.com

Abstract

A Study of BFLOAT16 for Deep Learning Training

Dhiraj Kalamkar[1], Dheevatsa Mudigere[2], Naveen Mellempudi*[1], Dipankar Das[1], Kunal Banerjee[1], Sasikanth Avancha[1], Dharma Teja Vooturi†[1], Nataraj Jammalamadaka†[1], Jianyu Huang[2], Hector Yuen[2], Jiyan Yang[2], Jongsoo Park[2], Alexander Heinecke[1], Evangelos Georganas[1], Sudarshan Srinivasan[1], Abhisek Kundu[1], Misha Smelyanskiy[2], Bharat Kaul[1], and Pradeep Dubey[1]
[1]Parallel Computing Lab, Intel Labs
[2]Facebook, 1 Hacker Way, Menlo Park, CA

Abstract

Ultra-Low Precision 4-bit Training of Deep Neural Networks

Xiao Sun, Naigang Wang, Chia-yu Chen, Jia-min Ni, Ankur Agrawal, Xiaodong Cui, Swagath Venkataramani, Kaoutar El Maghraoui, Vijayalakshmi Srinivasan, Kailash Gopalakrishnan
IBM T. J. Watson Research Center, Yorktown Heights, NY 10598, USA
{xsun, nwang, cchen, jiamin.ni, ankuragr, cuix, swagath.venkataramani, kelmaghr, viji, kailash}@us.ibm.com

Abstract

The 28th IEEE Int. Symp. on High-Performance Computer Architecture (HPCA-28), February 2022

FAST: DNN Training Under Variable Precision Block Floating Point with Stochastic Rounding

Sai Qian Zhang[1], Bradley McDanel[2], and H.T. Kung[1]
[1]Harvard University
[2]Franklin and Marshall College

FP8 FORMATS FOR DEEP LEARNING

Paulius Micikevicius, Dusan Stosic, Patrick Judd, John Kamalu, Stuart Oberman, Mohammad Shoeybi, Michael Siu, Hao Wu
NVIDIA
{pauliusm, dstosic, pjudd, jkamalu, soberman, mshoeybi, msiu, skyw}@nvidia.com
Neil Burgess, Sangwon Ha, Richard Grisenthwaite
Arm
{neil.burgess, sangwon.ha, richard.grisenthwaite}@arm.com
Naveen Mellempudi, Marius Cornea, Alexander Heinecke, Pradeep Dubey
Intel
{naveen.k.mellempudi, marius.cornea, alexander.heinecke, pradeep.dubey}@intel.com

ABSTRACT

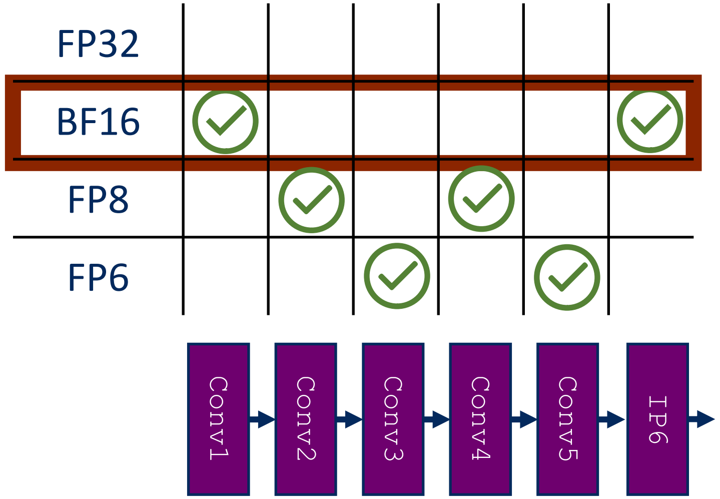
arXiv:2209.05433v2 [cs.LG] 29 Sep 2022

With Shared Microexponents, A Little Shifting Goes a Long Way

Bita Darvish Rouhani*, Ritchie Zhao, Venmugil Elango, Rasoul Shafipour, Mathew Hall, Maral Mesmakhosroshahi, Ankit More, Levi Melnick, Maximilian Golub, Girish Varatkar, Lei Shao, Gaurav Kolhe, Dimitry Melts, Jasmine Klar, Renee L'Heureux, Matt Perry, Doug Burger, Eric Chung
Microsoft
Zhaoxia (Summer) Deng, Sam Naghshineh, Jongsoo Park, Maxim Naumov
Meta

Which Datatype? Where? When?

- There are many great options
- But it's a trade-off
  - Accuracy vs. Efficiency
  - All have edge cases



	Conv1	Conv2	Conv3	Conv4	Conv5	IP6
FP32						
BF16	✓					✓
FP8		✓		✓		
FP6			✓		✓	

Published as a conference paper at ICLR 2018

MIXED PRECISION TRAINING

Sharan Narang, Gregory Diamos, Erich Elsen
Baidu Research
{sharan, gdiamos}@baidu.com

Paulius Micikevicius, Jonah Alben, David Garcia, Boris Ginsburg, Michael Houston, Oleksii Kuchaiev, Ganesh Venkatesh, Hao Wu
NVIDIA
{pauliusm, alben, dagarcia, bginsburg, mhouston, okuchaiev, gavenkatesh, skyw}@nvidia.com

Training DNNs with Hybrid Block Floating Point

Mario Drumond, Ecocloud, EPFL, mario.drumond@epfl.ch
Tao Lin, Ecocloud, EPFL, tao.lin@epfl.ch
Martin Jaggi, Ecocloud, EPFL, martin.jaggi@epfl.ch
Babak Falsafi, Ecocloud, EPFL, babak.falsafi@epfl.ch

Training Deep Neural Networks with 8-bit Floating Point Numbers

Naigang Wang, Jungwook Choi, Daniel Brand, Chia-Yu Chen and Kailash Gopalakrishnan
IBM T. J. Watson Research Center
Yorktown Heights, NY 10598, USA
{nwang, choij, danbrand, cchen, kailash}@us.ibm.com

A Study of BFLOAT16 for Deep Learning Training

Dhiraj Kalamkar, Dheevatsa Mudigere, Naveen Mellempudi, Dipankar Das, Kunal Banerjee, Sasikanth Avancha, Dharma Teja Vooturi, Nataraj Jammalamadaka, Jianyu Huang, Hector Yuen, Jiyan Yang, Jongsoo Park, Alexander Heinecke, Evangelos Georganas, Sudarshan Srinivasan, Abhisek Kundu, Misha Smelyanskiy, Bharat Kaul, and Pradeep Dubey
Parallel Computing Lab, Intel Labs
Facebook, 1 Hacker Way, Menlo Park, CA

Ultra-Low Precision 4-bit Training of Deep Neural Networks

Xiao Sun, Naigang Wang, Chia-yu Chen, Jia-min Ni, Ankur Agrawal, Xiaodong Cui, Swagath Venkataramani, Kaoutar El Maghraoui, Vijayalakshmi Srinivasan, Kailash Gopalakrishnan
IBM T. J. Watson Research Center, Yorktown Heights, NY 10598, USA

The 28th IEEE Int. Symp. on High-Performance Computer Architecture (HPCA-28), February 2022

FAST: DNN Training Under Variable Precision Block Floating Point with Stochastic Rounding

Sai Qian Zhang, Bradley McDanel, and H.T. Kung
Harvard University
Franklin and Marshall College

FP8 FORMATS FOR DEEP LEARNING

Paulius Micikevicius, Dusan Stosic, Patrick Judd, John Kamalu, Stuart Oberman, Mohammad Shoeybi, Michael Siu, Hao Wu
NVIDIA
Neil Burgess, Sangwon Ha, Richard Grisenthwaite
Arm
Naveen Mellempudi, Marius Cornea, Alexander Heinecke, Pradeep Dubey
Intel

arXiv:2209.05433v2 [cs.LG] 29 Sep 2022

With Shared Microexponents, A Little Shifting Goes a Long Way

Ritchie Zhao, Venmugil Elango, Rasoul Shafipour, Mathew Hall, Maral Mesmakhosroshahi, Levi Melnick, Maximilian Golub, Girish Varatkar, Lei Shao, Gaurav Kolhe, Dimitry Melts, Jasmine Klar, Renee L'Heureux, Matt Perry, Doug Burger, Eric Chung
Microsoft
Zhaoxia (Summer) Deng, Sam Naghshineh, Jongsoo Park, Maxim Naumov
Meta

Which Datatype? Where? When?

- There are many great options
- But it’s a trade-off
  - Accuracy vs. Efficiency
  - All have edge cases
- Guess
  - Confirm
  - Repeat

Published as a conference paper at ICLR 2018

MIXED PRECISION TRAINING

Sharan Narang*, Gregory Diamos, Erich Elsen†
Baidu Research
{sharan, gdiamos}@baidu.com

Paulius Micikevicius*, Jonah Alben, David Garcia, Boris Ginsburg, Michael Houston, Oleksii Kuchaiev, Ganesh Venkatesh, Hao Wu
NVIDIA
{pauliusm, alben, dagarcia, bginsburg, mhouston, okuchaiev, gavenkatesh, skyw}@nvidia.com

ABSTRACT

Training DNNs with Hybrid Block Floating Point

Mario Drumond
Ecocloud
EPFL
mario.drumond@epfl.ch

Tao Lin
Ecocloud
EPFL
tao.lin@epfl.ch

Martin Jaggi
Ecocloud
EPFL
martin.jaggi@epfl.ch

Babak Falsafi
Ecocloud
EPFL
babak.falsafi@epfl.ch

Abstract

The wide adoption of DNNs has given birth to … forcing datacenter operators to adopt … These accelerators typically employ … arithmetic to maximize perf… further increase that per… point arithmetic. H… point's narro… We ide…

Training Deep Neural Networks with 8-bit Floating Point Numbers

Naigang Wang, Jungwook Choi, Daniel Brand, Chia-Yu Chen and Kailash Gopalakrishnan
IBM T. J. Watson Research Center
Yorktown Heights, NY 10598, USA
{nwang, choij, danbrand, cchen, kailash}@us.ibm.com

Abstract

The state-of-the-art hardware platforms for training Deep Neural Networks (DNNs) are moving from traditional single precision (32-bit) computations … bits of precision – in large part due to the high energy … bit storage associated with using reduced-precision … unlike inference, training with numbers … challenging due to the need … back-propagation … of DNN…

A Study of BFLOAT16 for Deep Learning Training

Dhiraj Kalamkar[1], Dheevatsa Mudigere[2], Naveen Mellempudi*[1], Dipankar Das[1], Kunal Banerjee[1], Sasikanth Avancha[1], Dharma Teja Vooturi†[1], Nataraj Jammalamadaka†[1], Jianyu Huang[2], Hector Yuen[2], Jiyan Yang[2], Jongsoo Park[2], Alexander Heinecke[1], Evangelos Georganas[1], Sudarshan Srinivasan[1], Abhisek Kundu[1], Misha Smelyanskiy[2], Bharat Kaul[1], and Pradeep Dubey[1]

[1]Parallel Computing Lab, Intel Labs
[2]Facebook, 1 Hacker Way, Menlo Park, CA

Abstract

This paper presents the first comprehensive empirical study … efficacy of the Brain Floating Point (BF16) … Learning training …

Ultra-Low Precision 4-bit Training of Deep Neural Networks

Xiao Sun, Naigang Wang, Chia-yu Chen, Jia-min Ni, Ankur Agrawal, Xiaodong Cui, Swagath Venkataramani, Kaoutar El Maghraoui, Vijayalakshmi Srinivasan, Kailash Gopalakrishnan
IBM T. J. Watson Research Center, Yorktown Heights, NY 10598, USA
{xsun, nwang, cchen, jiamin.ni, ankuragr, cuix, kelmaghr, viji,kailash}@us.ibm.com
swagath.venkataramani

Abstract

… of novel techniques and numerical repres… time, the precision of training systems … advance … To enable this … (GradScale) that addres… gradients as … model training …

The 28th IEEE Int. Symp. on High-Performance Computer Architecture (HPCA-28), February 2022

FAST: DNN Training Under Variable Precision Block Floating Point with Stochastic Rounding

Sai Qian Zhang[1], Bradley McDanel[2], and H.T. Kung[1]
[1]Harvard University
[2]Franklin and Marshall College

Abstract—Block Floating Point (BFP) can efficiently support quantization for Deep Neural Network (DNN) training by providing a wide dynamic range via a shared exponent across a group of values. In this paper, we propose a Fast First, Accurate Second Training (FAST) system for DNNs, where the weights, activations, and gradients are represented in BFP. FAST supports matrix multiplication with variable precision BFP input operands, enabling incremental increases in DNN precision throughout training. By increasing the BFP precision across both training iterations and DNN layers, FAST can greatly shorten the training time while reducing overall hardware resource usage. Our FAST Multiplier-Accumulator (fMAC) supports dot product computations under multiple BFP precisions. We validate our FAST system on multiple DNNs with different datasets, demonstrating a 2-6× speedup in training on a single-chip platform over prior work based on mixed-precision or block floating point number systems while achieving similar performance in validation accuracy.

Fig. 1. (left) Overview of the proposed FAST system for DNN training … variable precision BFP with stochastic rounding. A group of FP values … converted to a BFP group with a selected mantissa width based on … error tolerance (see Algorithm 1). (right) FAST incrementally increases … BFP precision across both layer depths and training iterations … on corresponding empirical data obtained via DNN training.

FP8 FORMATS FOR DEEP LEARNING

Paulius Micikevicius, Dusan Stosic, Patrick Judd, John Kamalu, Stuart Oberman, Mohammad Shoeybi, Michael Siu, Hao Wu
NVIDIA
{pauliusm, dstosic, pjudd, jkamalu, soberman, mshoeybi, msiu, skyw}@nvidia.com

Neil Burgess, Sangwon Ha, Richard Grisenthwaite
Arm
{neil.burgess, sangwon.ha, richard.grisenthwaite}@arm.com

Naveen Mellempudi, Marius Cornea, Alexander Heinecke, Pradeep Dubey
Intel
{naveen.k.mellempudi, marius.cornea, alexander.heinecke, pradeep.dubey}@intel.com

ABSTRACT

FP8 is a natural progression for accelerating deep learning training inference … formats common in modern processors. In this paper we propose an 8-bit FP8 … interchange format consisting of two encodings - E4M3 (4-bit exponent and 3-bit mantissa) and E5M2 (5-bit exponent and 2-bit mantissa). While E5M2 … special values, E4M3's dynamic range … mantissa bit-pattern for NaNs … and language tasks … study cover…

arXiv:2209.05433v2 [cs.LG] 29 Sep 2022

With Shared Microexponents, A Little Shifting Goes a Long Way

…uhani*, Ritchie Zhao, Venmugil Elango, Rasoul Shafipour, Mathew Hall, Maral Mesmakhosroshahi, …More, Levi Melnick, Maximilian Golub, Girish Varatkar, Lei Shao, Gaurav Kolhe, Dimitry Melts, Jasmine Klar, Renee L'Heureux, Matt Perry, Doug Burger, Eric Chung
Microsoft
Zhaoxia (Summer) Deng, Sam Naghshineh, Jongsoo Park, Maxim Naumov
Meta

…stract—This paper introduces Block Data Representations (BDR), a framework for exploring and evaluating a wide spectrum of narrow-precision formats for deep learning. It enables comparison of popular quantization standards, and … BDR, new formats based on …

Many data formats have been proposed, and the … space of possible new ideas …

Which Datatype? Where? When?

- There are many great options
- But it’s a trade-off
  - Accuracy vs. Efficiency
  - All have edge cases
- Guess
  - Confirm
  - Repeat
- Lost Potential

Published as a conference paper at ICLR 2018

MIXED PRECISION TRAINING

Sharan Narang*, Gregory Diamos, Erich Elsen†
Baidu Research
{sharan, gdiamos}@baidu.com

Paulius Micikevicius*, Jonah Alben, David Garcia, Boris Ginsburg, Michael Houston, Oleksii Kuchaiev, Ganesh Venkatesh, Hao Wu
NVIDIA
{pauliusm, alben, dagarcia, bginsburg, mhouston, okuchaiev, gavenkatesh, skyw}@nvidia.com

Training DNNs with Hybrid Block Floating Point

Mario Drumond, Ecocloud, EPFL, mario.drumond@epfl.ch
Tao Lin, Ecocloud, EPFL, tao.lin@epfl.ch
Martin Jaggi, Ecocloud, EPFL, martin.jaggi@epfl.ch
Babak Falsafi, Ecocloud, EPFL, babak.falsafi@epfl.ch

Training Deep Neural Networks with 8-bit Floating Point Numbers

Naigang Wang, Jungwook Choi, Daniel Brand, Chia-Yu Chen and Kailash Gopalakrishnan
IBM T. J. Watson Research Center
Yorktown Heights, NY 10598, USA
{nwang, choij, danbrand, cchen, kailash}@us.ibm.com

A Study of BFLOAT16 for Deep Learning Training

Dhiraj Kalamkar[1], Dheevatsa Mudigere[2], Naveen Mellempudi*[1], Dipankar Das[1], Kunal Banerjee[1], Sasikanth Avancha[1], Dharma Teja Vooturi†[1], Nataraj Jammalamadaka†[1], Jianyu Huang[2], Hector Yuen[2], Jiyan Yang[2], Jongsoo Park[2], Alexander Heinecke[1], Evangelos Georganas[1], Sudarshan Srinivasan[1], Abhisek Kundu[1], Misha Smelyanskiy[2], Bharat Kaul[1], and Pradeep Dubey[1]
[1]Parallel Computing Lab, Intel Labs
[2]Facebook, 1 Hacker Way, Menlo Park, CA

Ultra-Low Precision 4-bit Training of Deep Neural Networks

Xiao Sun, Naigang Wang, Chia-yu Chen, Jia-min Ni, Ankur Agrawal, Xiaodong Cui, Swagath Venkataramani, Kaoutar El Maghraoui, Vijayalakshmi Srinivasan, Kailash Gopalakrishnan
IBM T. J. Watson Research Center, Yorktown Heights, NY 10598, USA

The 28th IEEE Int. Symp. on High-Performance Computer Architecture (HPCA-28), February 2022

FAST: DNN Training Under Variable Precision Block Floating Point with Stochastic Rounding

Sai Qian Zhang[1], Bradley McDanel[2], and H.T. Kung[1]
[1]Harvard University
[2]Franklin and Marshall College

FP8 FORMATS FOR DEEP LEARNING

Paulius Micikevicius, Dusan Stosic, Patrick Judd, John Kamalu, Stuart Oberman, Mohammad Shoeybi, Michael Siu, Hao Wu
NVIDIA
Neil Burgess, Sangwon Ha, Richard Grisenthwaite
Arm
Naveen Mellempudi, Marius Cornea, Alexander Heinecke, Pradeep Dubey
Intel

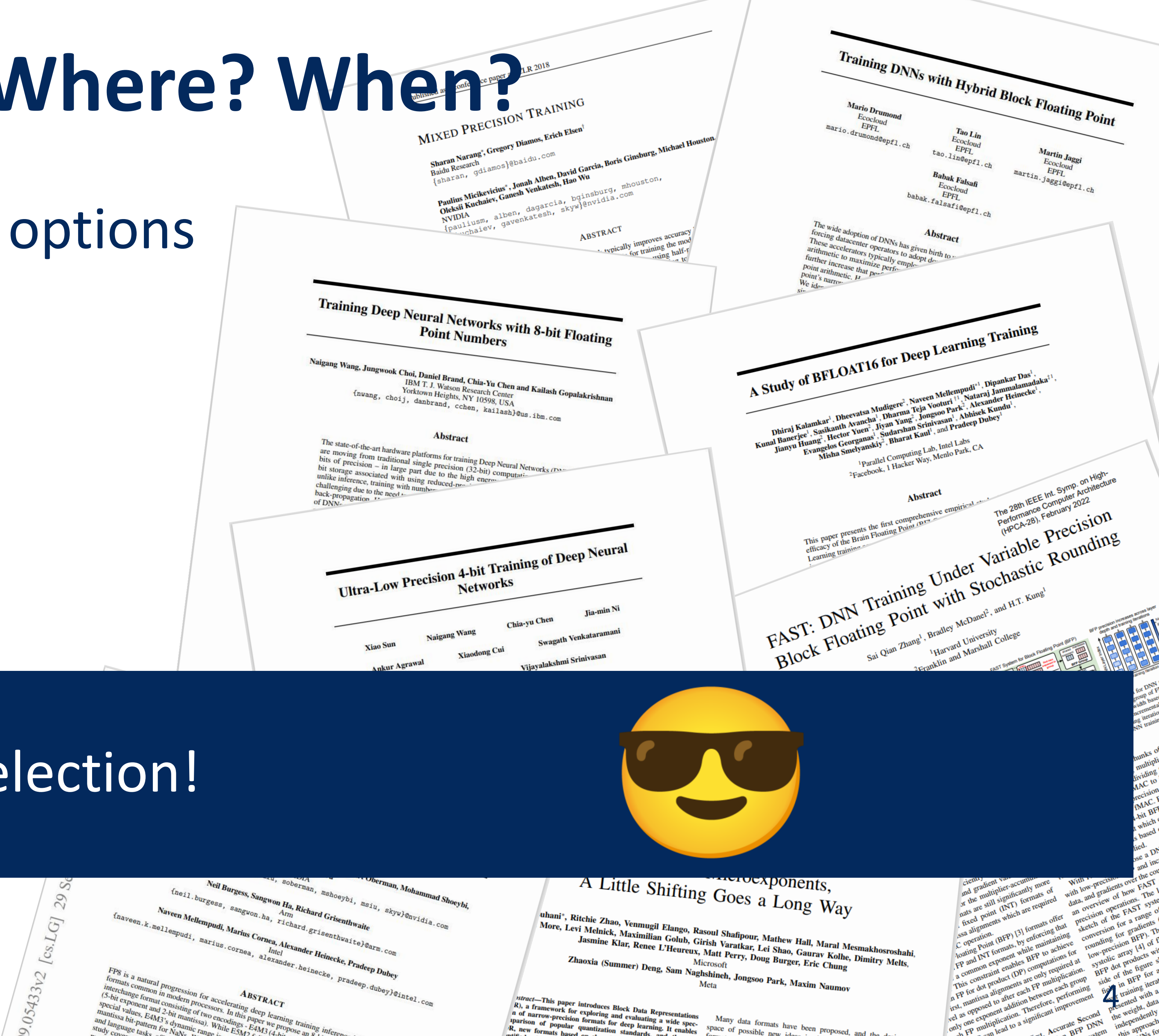
arXiv:2209.05433v2 [cs.LG] 29 Sep 2022

With Shared Microexponents, A Little Shifting Goes a Long Way

Microsoft
Meta

Which Datatype? Where? When?

- There are many great options
- But it’s a trade-off
  - Accuracy vs. Efficiency
  - All have edge cases
- Guess
  - Confirm
  - Repeat



Published as a conference paper at ICLR 2018

MIXED PRECISION TRAINING

Sharan Narang*, Gregory Diamos, Erich Elsen†
Baidu Research
{sharan, gdiamos}@baidu.com

Paulius Micikevicius*, Jonah Alben, David Garcia, Boris Ginsburg, Michael Houston, Oleksii Kuchaiev, Ganesh Venkatesh, Hao Wu
NVIDIA
{pauliusm, alben, dagarcia, bginsburg, mhouston, okuchaiev, gavenkatesh, skyw}@nvidia.com

ABSTRACT

Training DNNs with Hybrid Block Floating Point

Mario Drumond, Ecocloud, EPFL, mario.drumond@epfl.ch
Tao Lin, Ecocloud, EPFL, tao.lin@epfl.ch
Martin Jaggi, Ecocloud, EPFL, martin.jaggi@epfl.ch
Babak Falsafi, Ecocloud, EPFL, babak.falsafi@epfl.ch

Abstract

Training Deep Neural Networks with 8-bit Floating Point Numbers

Naigang Wang, Jungwook Choi, Daniel Brand, Chia-Yu Chen and Kailash Gopalakrishnan
IBM T. J. Watson Research Center
Yorktown Heights, NY 10598, USA
{nwang, choij, danbrand, cchen, kailash}@us.ibm.com

Abstract

A Study of BFLOAT16 for Deep Learning Training

Dhiraj Kalamkar[1], Dheevatsa Mudigere[2], Naveen Mellempudi*[1], Dipankar Das[1], Kunal Banerjee[1], Sasikanth Avancha[1], Dharma Teja Vooturi†[1], Nataraj Jammalamadaka[1], Jianyu Huang[2], Hector Yuen[2], Jiyan Yang[2], Jongsoo Park[2], Alexander Heinecke[1], Evangelos Georganas[1], Sudarshan Srinivasan[1], Abhisek Kundu[1], Misha Smelyanskiy[2], Bharat Kaul[1], and Pradeep Dubey[1]
[1]Parallel Computing Lab, Intel Labs
[2]Facebook, 1 Hacker Way, Menlo Park, CA

Abstract

Ultra-Low Precision 4-bit Training of Deep Neural Networks

Xiao Sun, Naigang Wang, Chia-yu Chen, Jia-min Ni, Ankur Agrawal, Xiaodong Cui, Swagath Venkataramani, Vijayalakshmi Srinivasan

The 28th IEEE Int. Symp. on High-Performance Computer Architecture (HPCA-28), February 2022

FAST: DNN Training Under Variable Precision Block Floating Point with Stochastic Rounding

Sai Qian Zhang[1], Bradley McDanel[2], and H.T. Kung[1]
[1]Harvard University
[2]Franklin and Marshall College

A Little Shifting Goes a Long Way

Neil Burgess, Sangwon Ha, Richard Grisenthwaite
Arm
Naveen Mellempudi, Marius Cornea, Alexander Heinecke, Pradeep Dubey
Intel

ABSTRACT

Automate the datatype selection! 😎

Our Goal

- Oracle?

Our Goal

- Oracle?
- Learn the Datatype
  - Use gradient descent

Our Goal

- Oracle?
- Learn the Datatype
  - Use gradient descent
- Monitor Loss
  - Adapt accordingly

Our Goal

- Oracle?
- Learn the Datatype
  - Use gradient descent
- Monitor Loss
  - Adapt accordingly
- Result
  - Less bits
  - Less energy
  - Less time
  - Bigger models

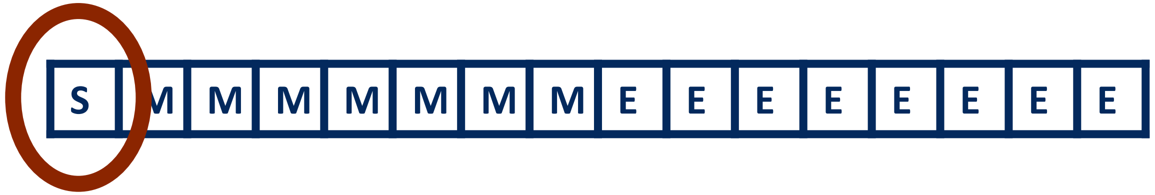
Datatype – Floating Point

- $Value = (-1)^{S} \times M \times 2^{E}$

S	M	M	M	M	M	M	M	E	E	E	E	E	E	E	E

Datatype – Floating Point

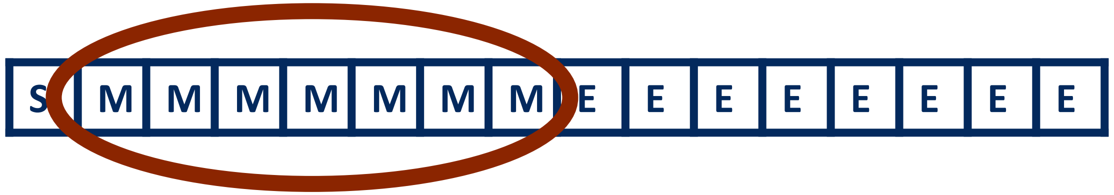
- $Value = (-1)^S \times M \times 2^E$
- Sign is trivial



S	M	M	M	M	M	M	M	E	E	E	E	E	E	E	E

Datatype – Floating Point

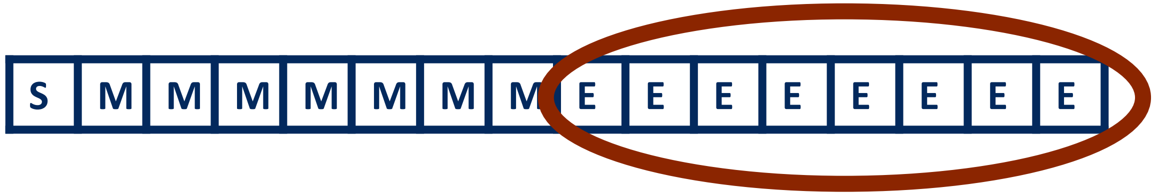
- $Value = (-1)^S \times M \times 2^E$
- Sign is trivial
- Mantissa
  - Precision



S	M	M	M	M	M	M	M	E	E	E	E	E	E	E	E

Datatype – Floating Point

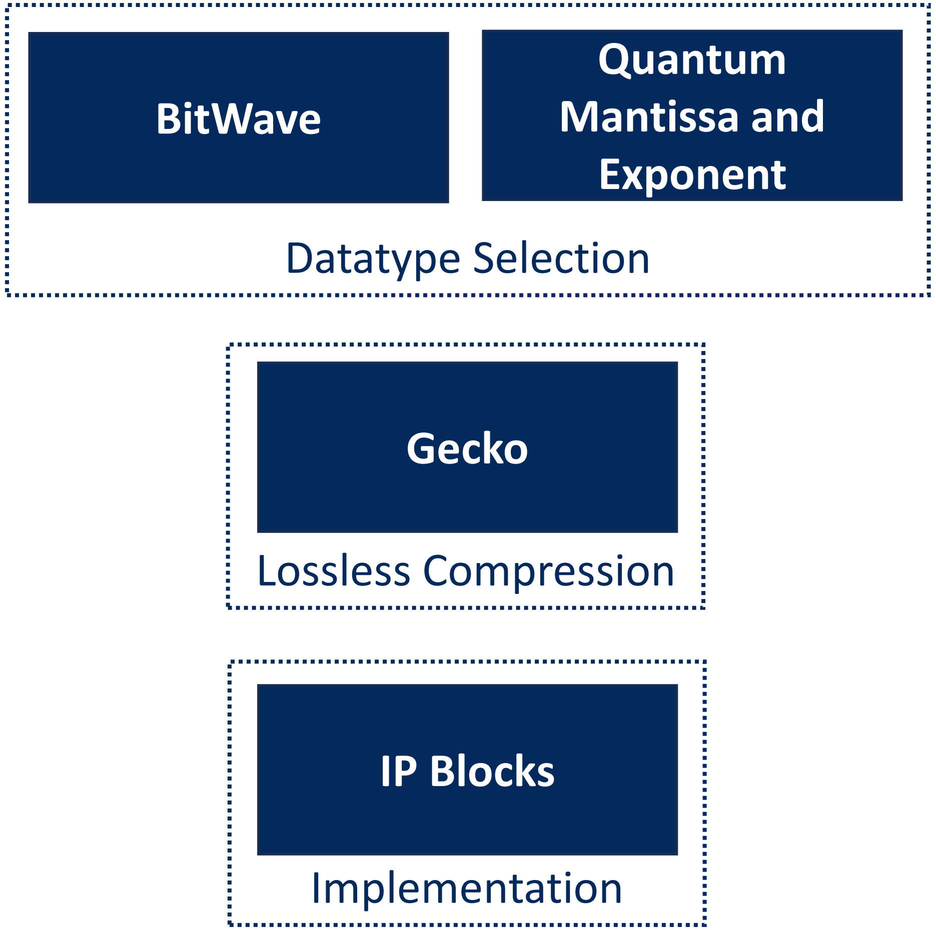
- $Value = (-1)^S \times M \times 2^E$
- Sign is trivial
- Mantissa
  - Precision
- Exponent
  - Range



S	M	M	M	M	M	M	M	E	E	E	E	E	E	E	E

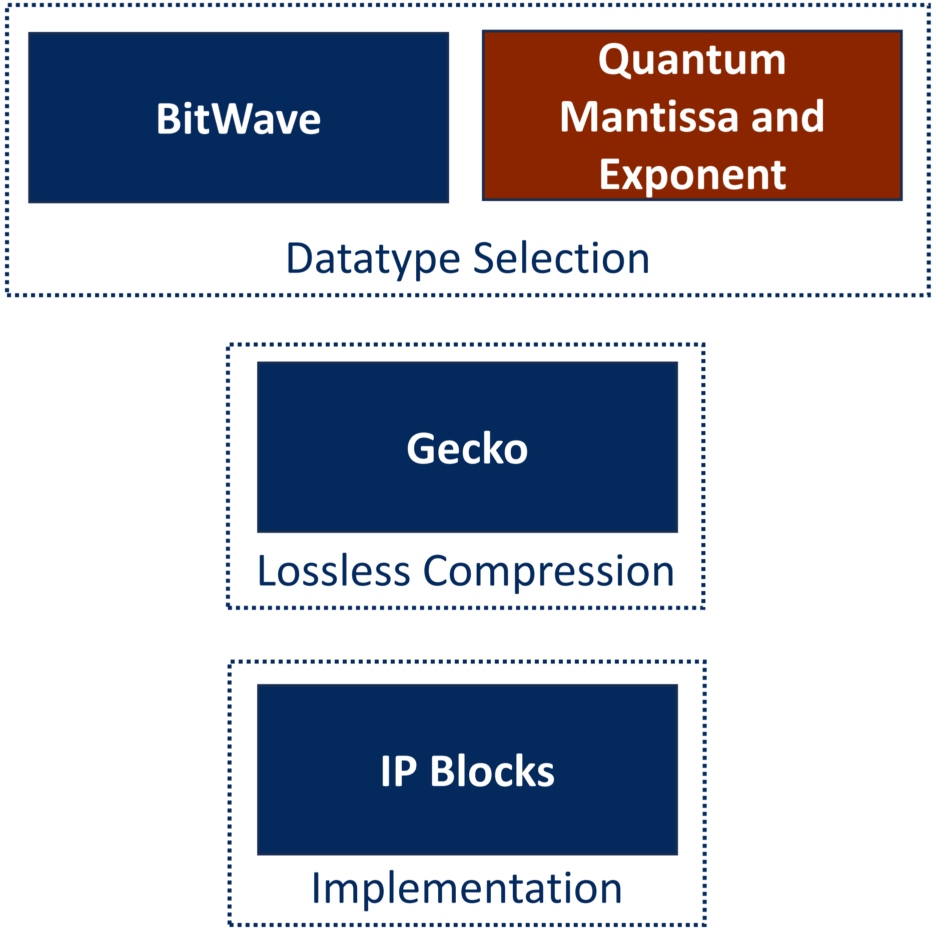
Schrödinger’s FP

- Machine Learning
  - *Quantum Mantissa and Exponent* (**4.7x** reduction)
- Black Box Sampling
  - *BitWave* (**3.2x** reduction)
- Exponent compression
  - *Gecko* (boost to **5.6x** and **4.6x**)
- Hardware IP blocks



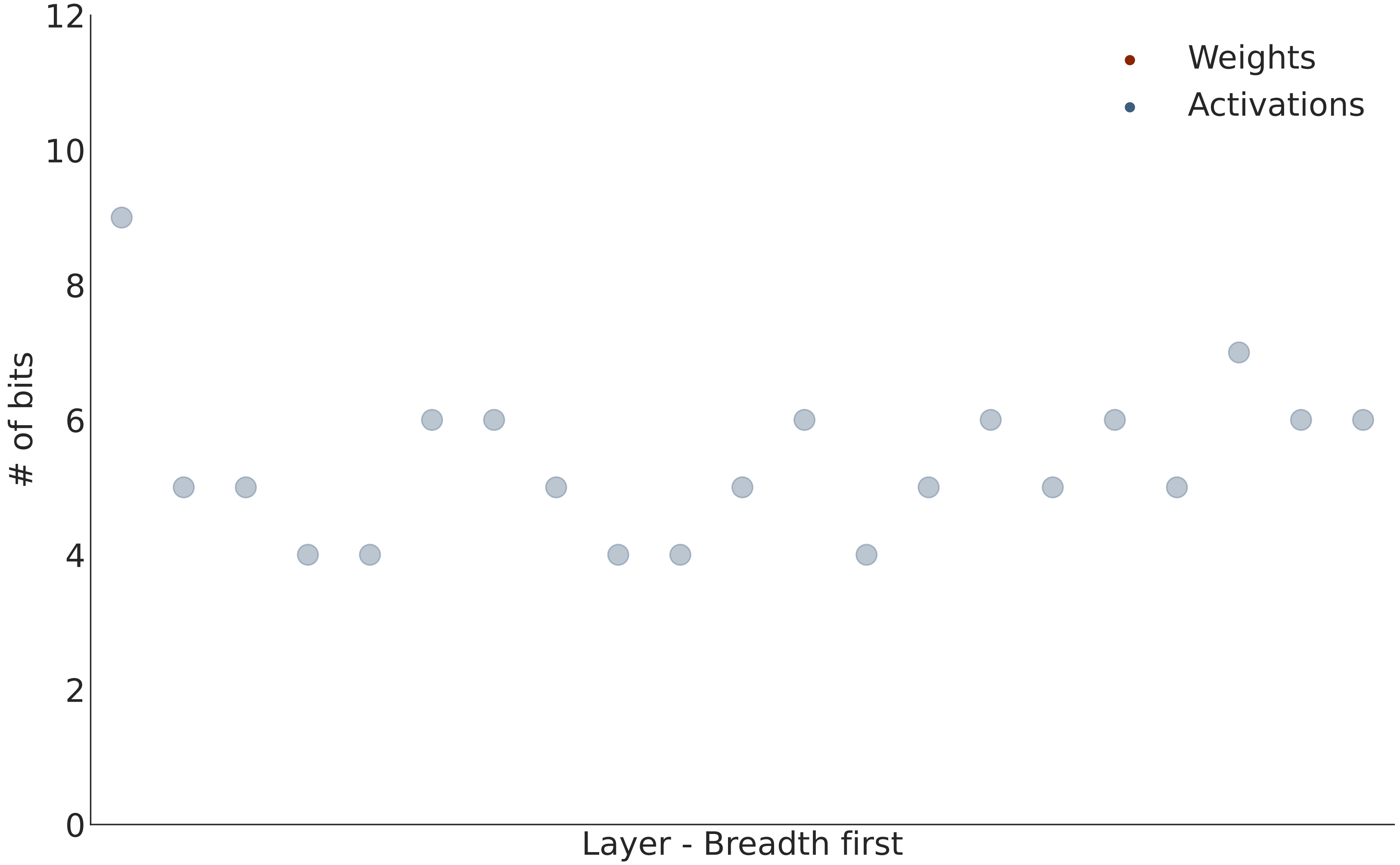
Schrödinger’s FP

- Machine Learning
  - *Quantum Mantissa and Exponent*
- Black Box Sampling
  - *BitWave*
- Exponent compression
  - *Gecko*
- Hardware IP blocks



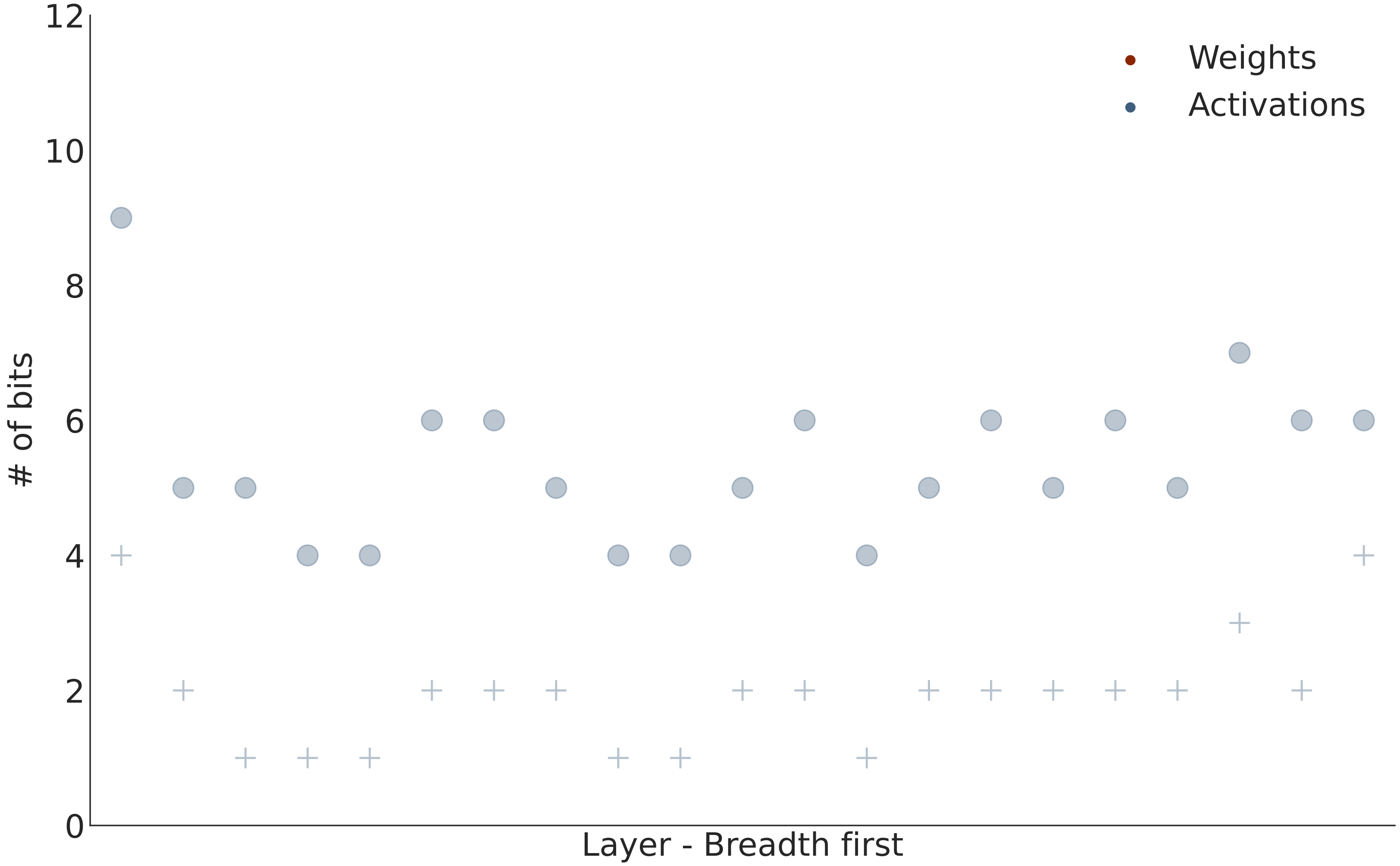
Per Tensor Datatype

- ResNet18
  - Per Tensor?
  - Single datatype?



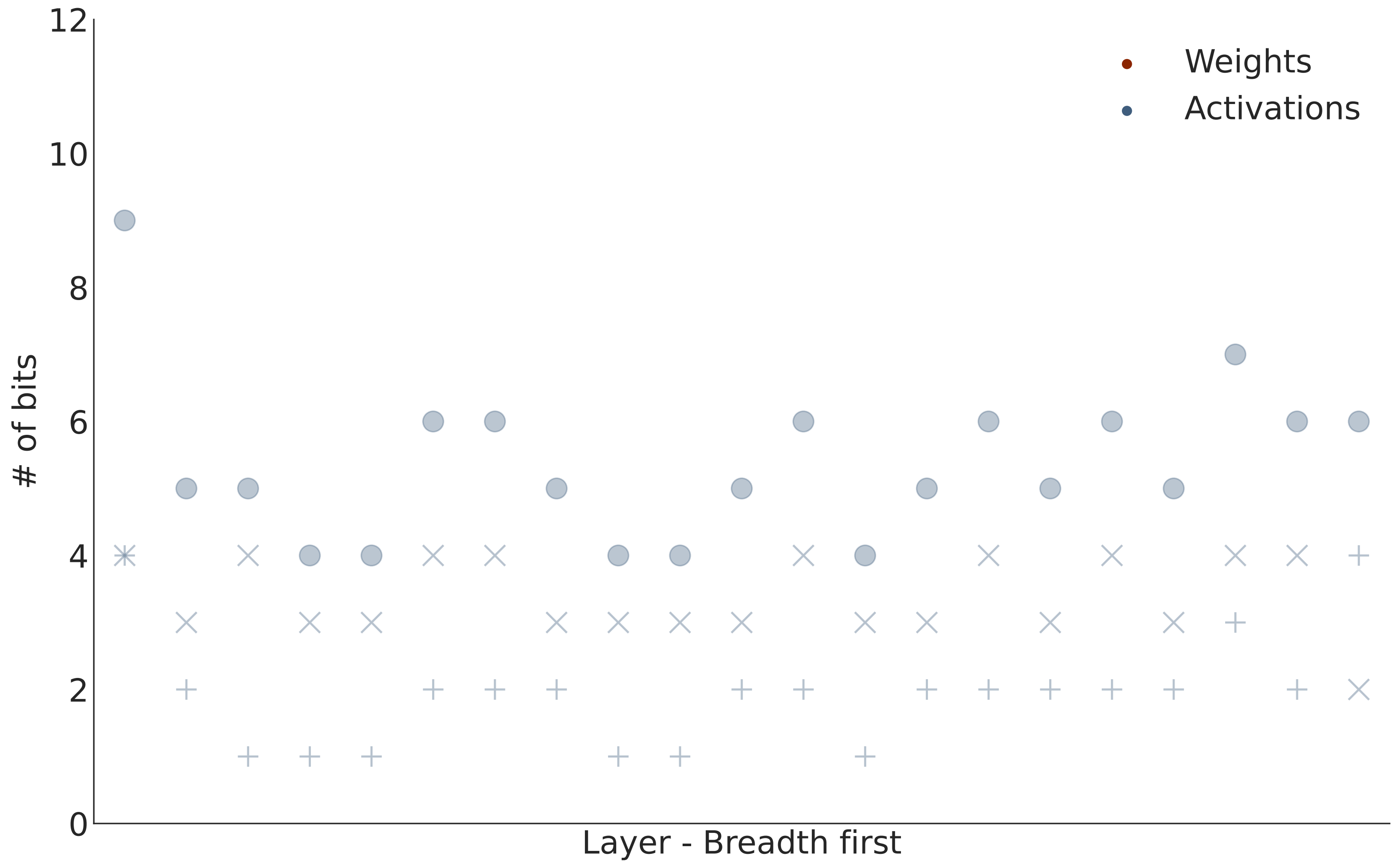
Per Tensor Datatype

- ResNet18
  - Per Tensor?
  - Single datatype?



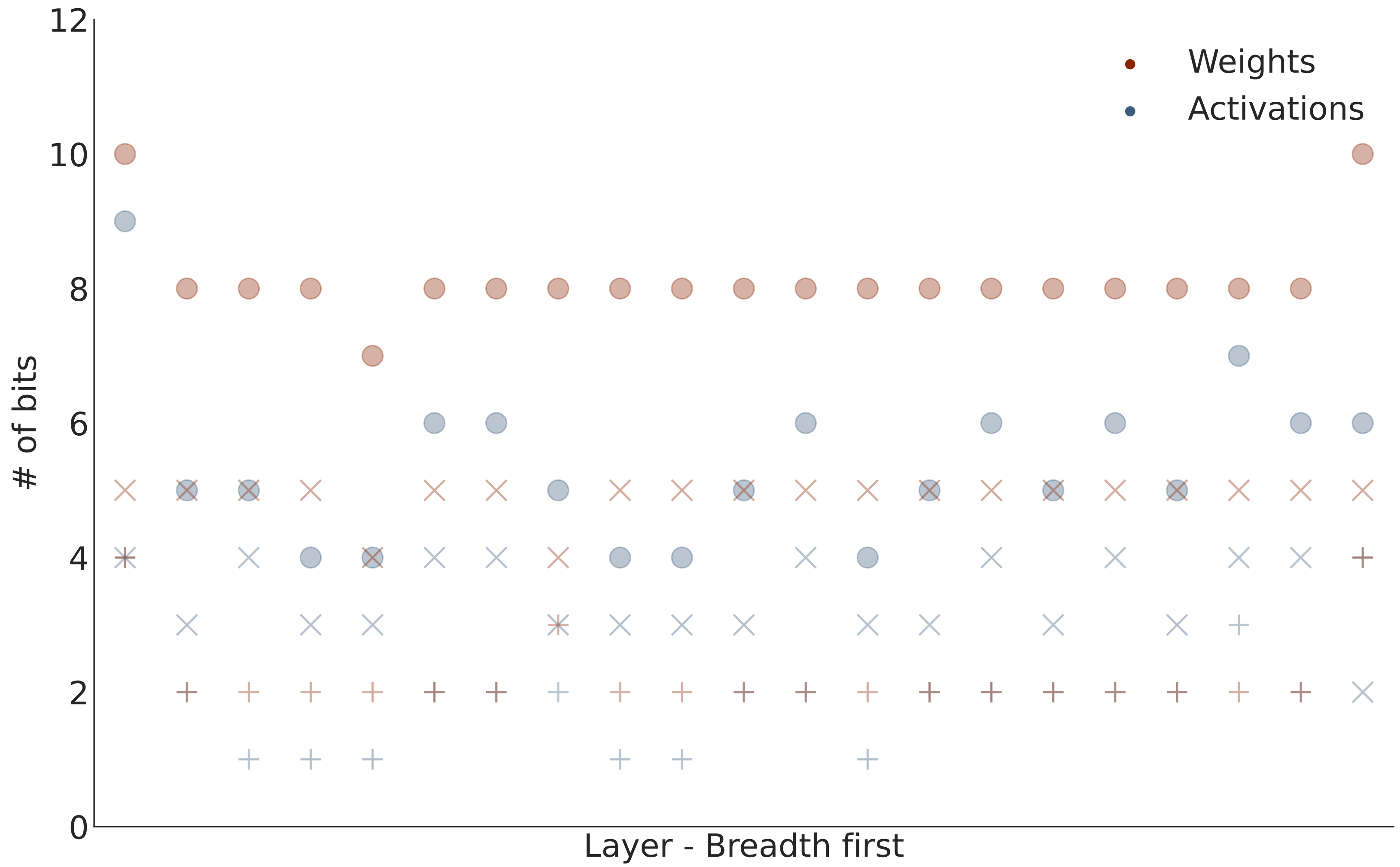
Per Tensor Datatype

- ResNet18
  - Per Tensor?
  - Single datatype?



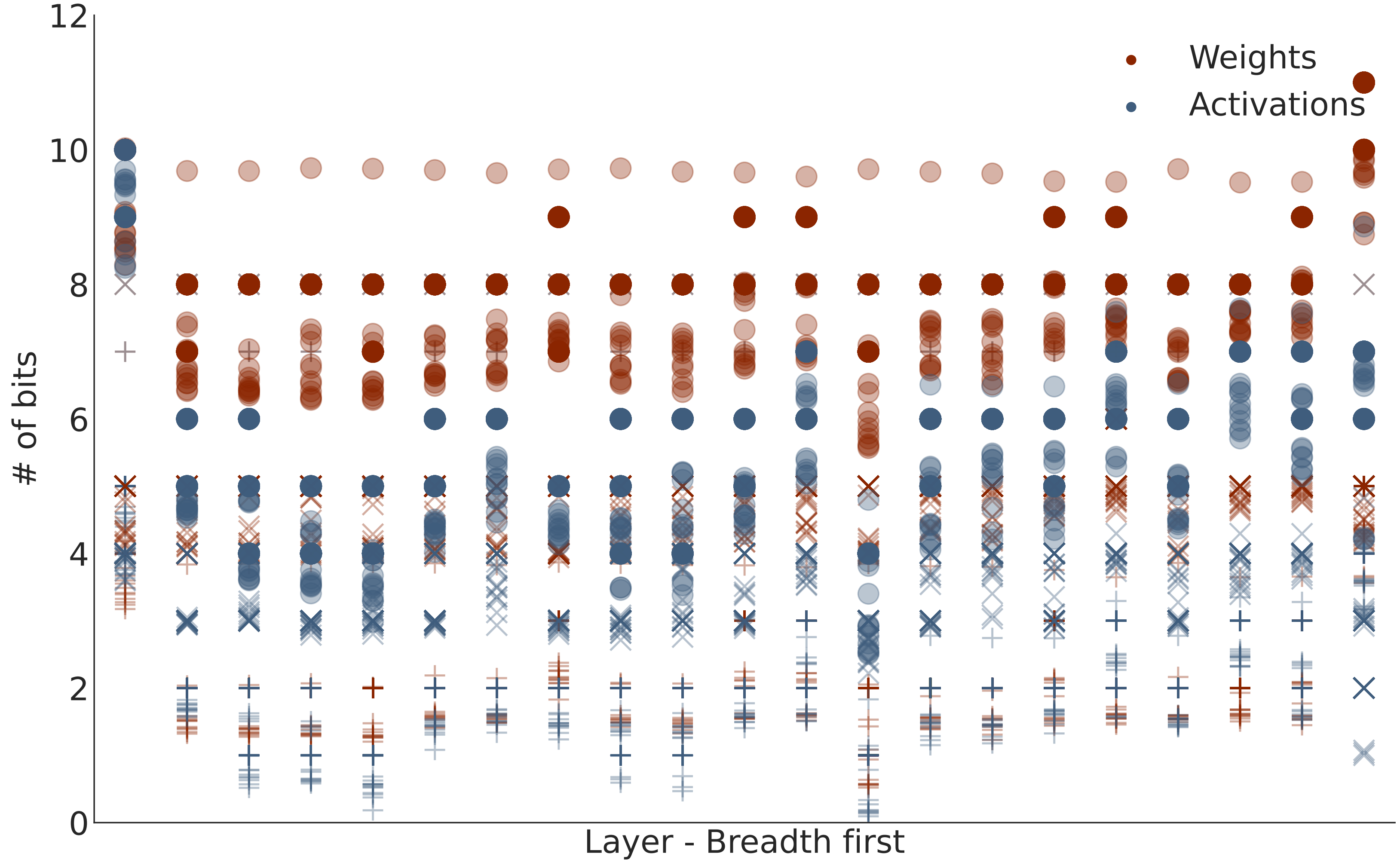
Per Tensor Datatype

- ResNet18
  - Per Tensor?
  - Single datatype?



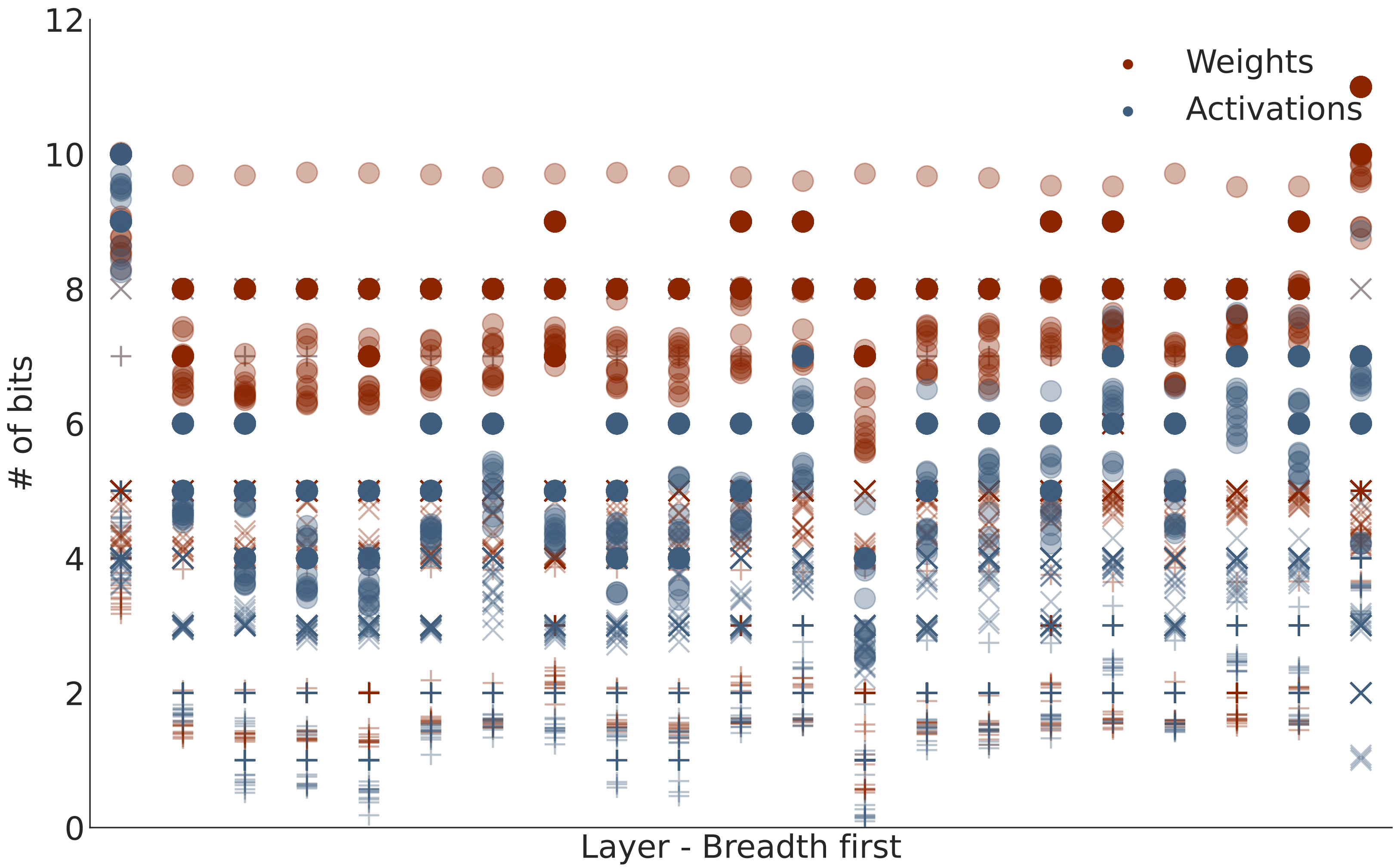
Per Tensor Datatype

- ResNet18
  - Per Tensor?
  - Single datatype?



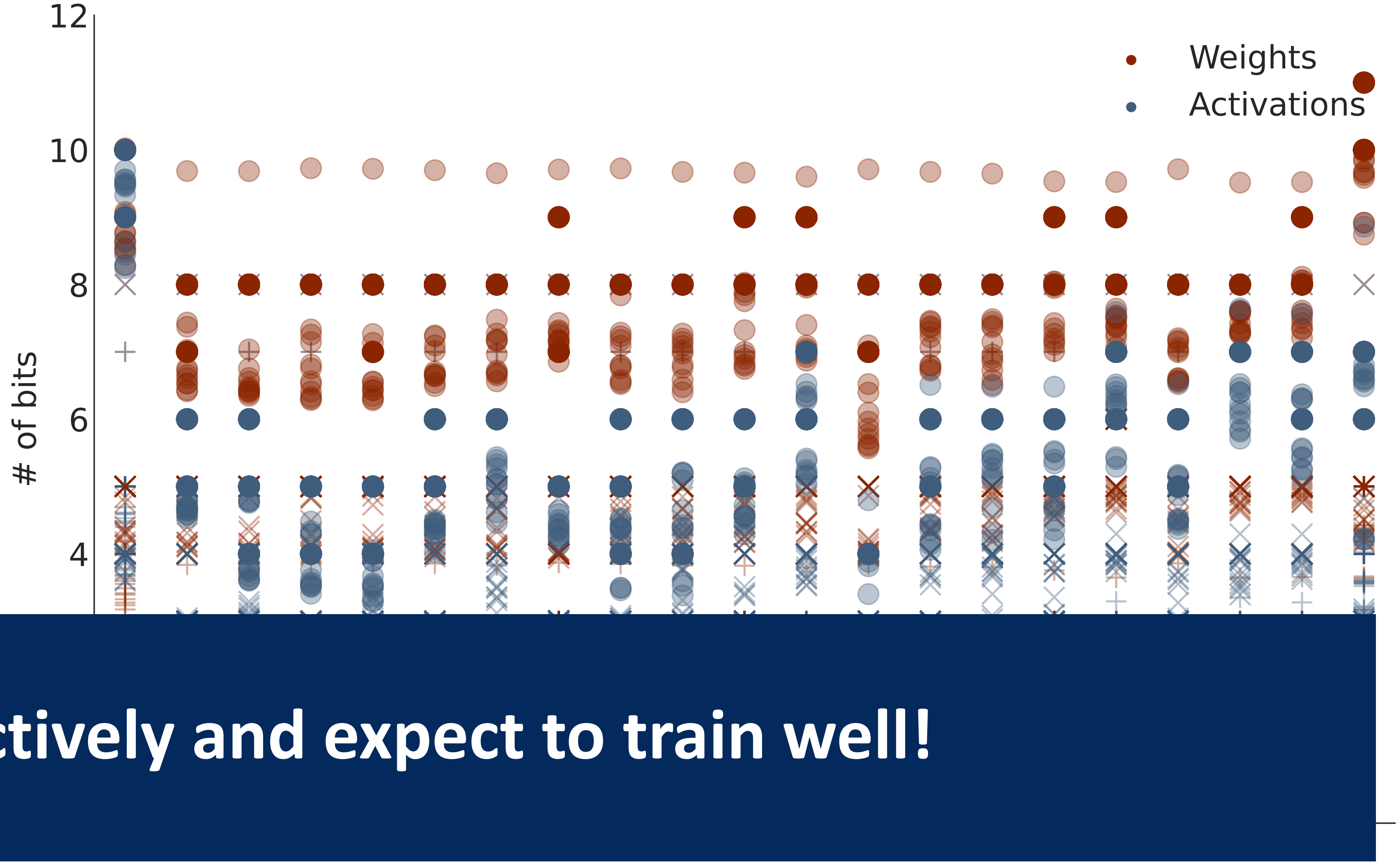
Per Tensor Datatype

- ResNet18
    - Per Tensor?
    - Single datatype?
- Less than 6b on average!
    - Much better than network wide!



Per Tensor Datatype

- ResNet18
  - Per Tensor?
  - Single datatype?
- Less than 6b on average!
  - Much better than network wide!



No chance to guess effectively and expect to train well!

Quantum Mantissa Loss

- Our Loss
  - $L = L_O + \gamma \times (\sum \lambda_i \times \alpha_i)$
  - Weighted according to priority
    - Tensor footprint
    - # of operations

Quantum Mantissa Loss

Original Loss

- Our Loss
  - $L = L_O + \gamma \times (\sum \lambda_i \times \alpha_i)$
  - Weighted according to priority
    - Tensor footprint
    - # of operations

Quantum Mantissa Loss

Mantissa Lengths

- Our Loss
  - $L = L_O + \gamma \times (\sum \lambda_i \times \alpha_i)$
  - Weighted according to priority
    - Tensor footprint
    - # of operations

Quantum Mantissa Loss

- Our Loss

  Mantissa Weights

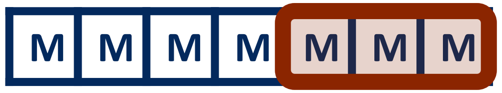
  - $L = L_O + \gamma \times (\sum \lambda_i \times \alpha_i)$
  - Weighted according to priority
    - Tensor footprint
    - # of operations

Quantum Mantissa Loss

- Our Loss
  - $L = L_O + \underset{}{\overset{\textbf{Regularizer Strength}}{\gamma}} \times (\sum \lambda_i \times \alpha_i)$
  - Weighted according to priority
    - Tensor footprint
    - # of operations

Quantum Mantissa Bitlength

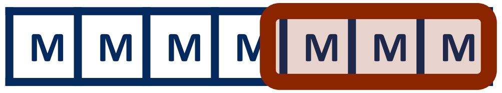
- Int Datatype



M M M M M M M

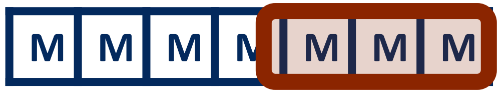
Quantum Mantissa Bitlength

- Int Datatype
- Non-Int Datatype



Quantum Mantissa Bitlength

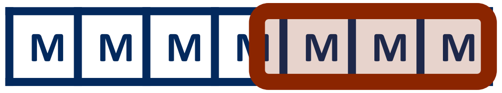
- Int Datatype
- Non-Int Datatype



M M M M M M M

Quantum Mantissa Bitlength

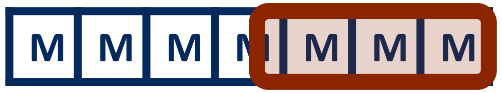
- Int Datatype
- Non-Int Datatype

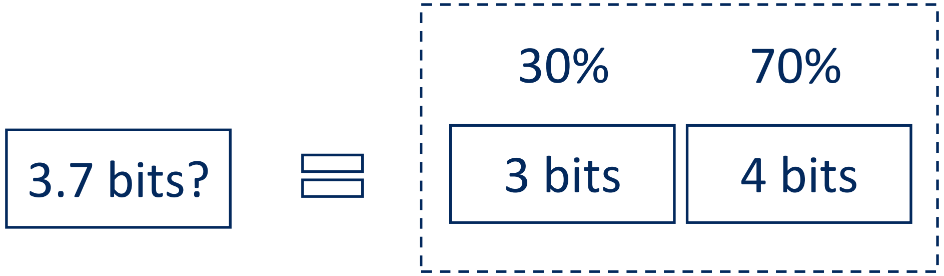


M M M M M M M

Quantum Mantissa Bitlength

- Int Datatype
- Non-Int Datatype




3.7 bits?

≡

30%

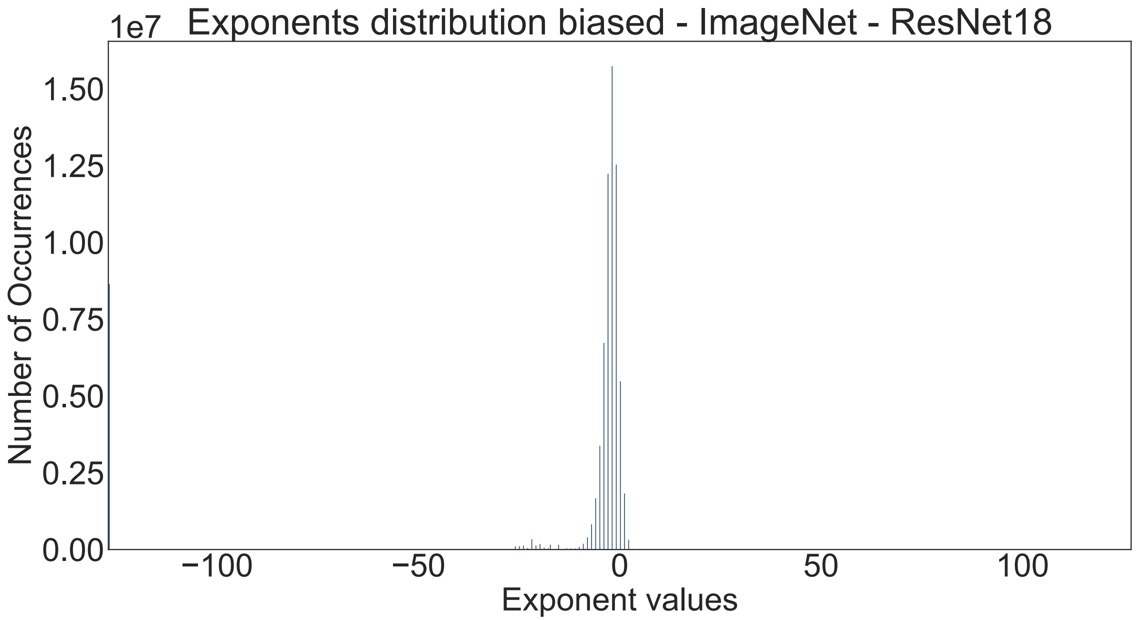
70%

3 bits

4 bits

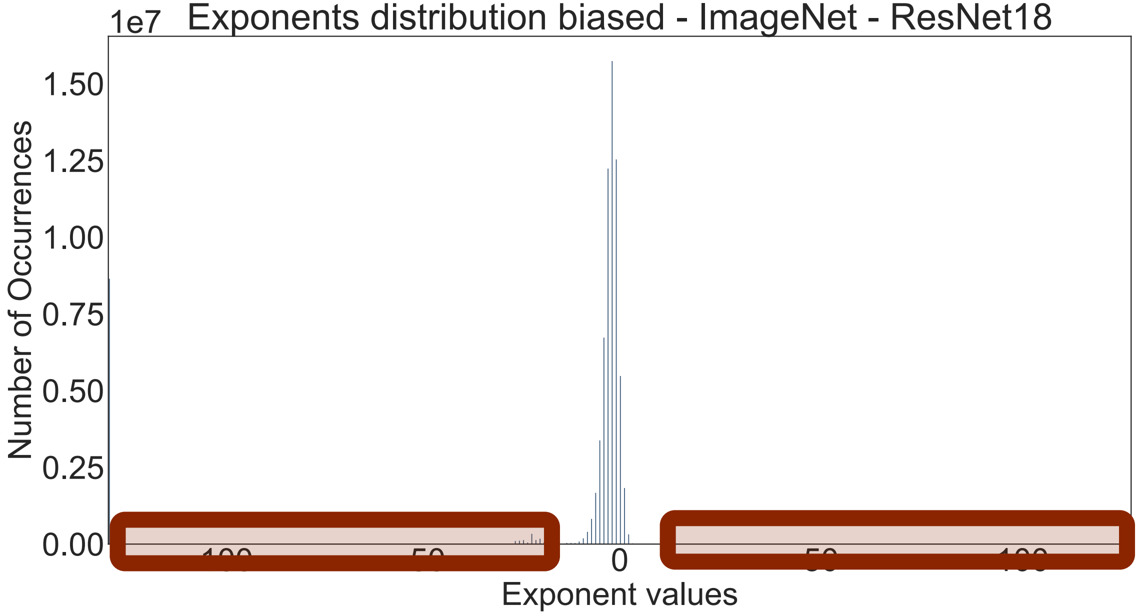
Quantum Exponent Range

- Distribution



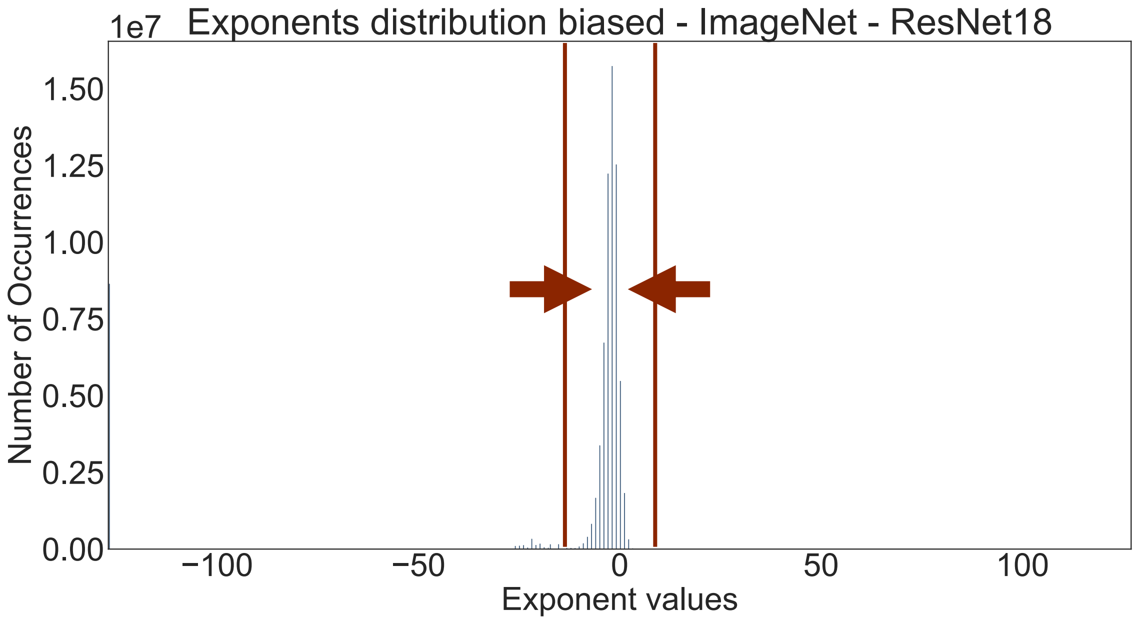
Quantum Exponent Range

- Distribution



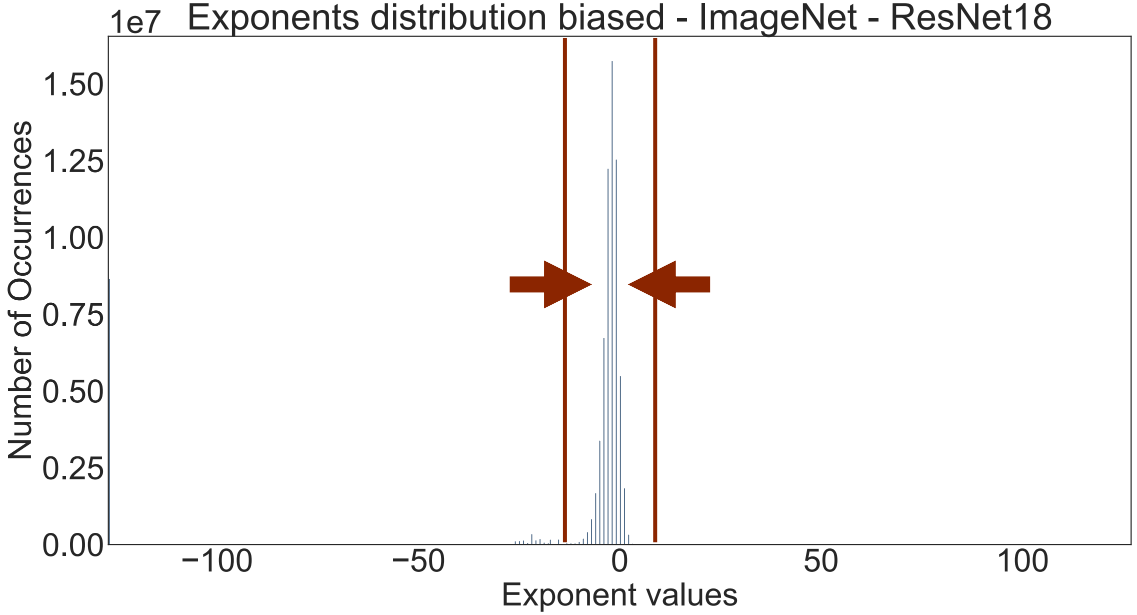
Quantum Exponent Range

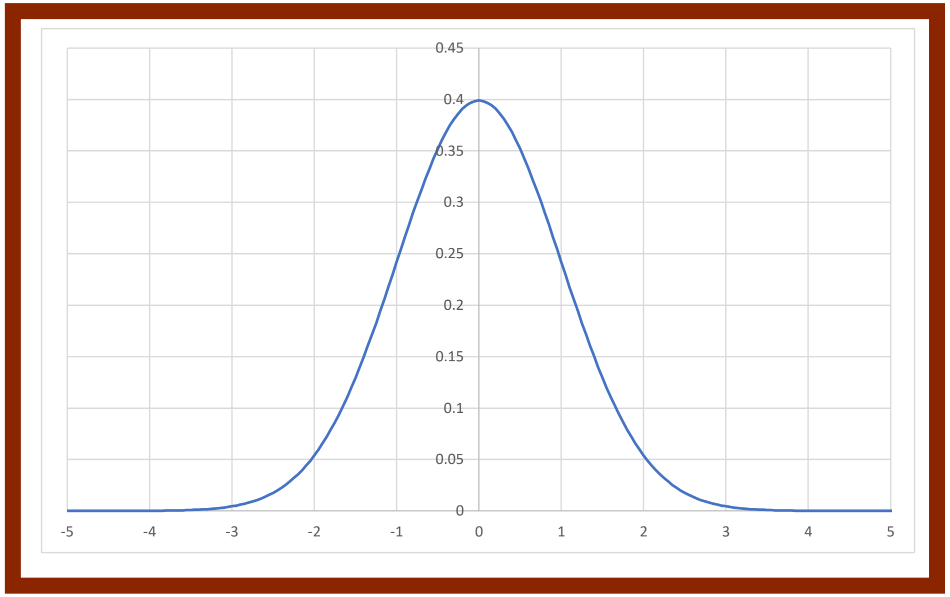
- Distribution
- Compress range



Quantum Exponent Range

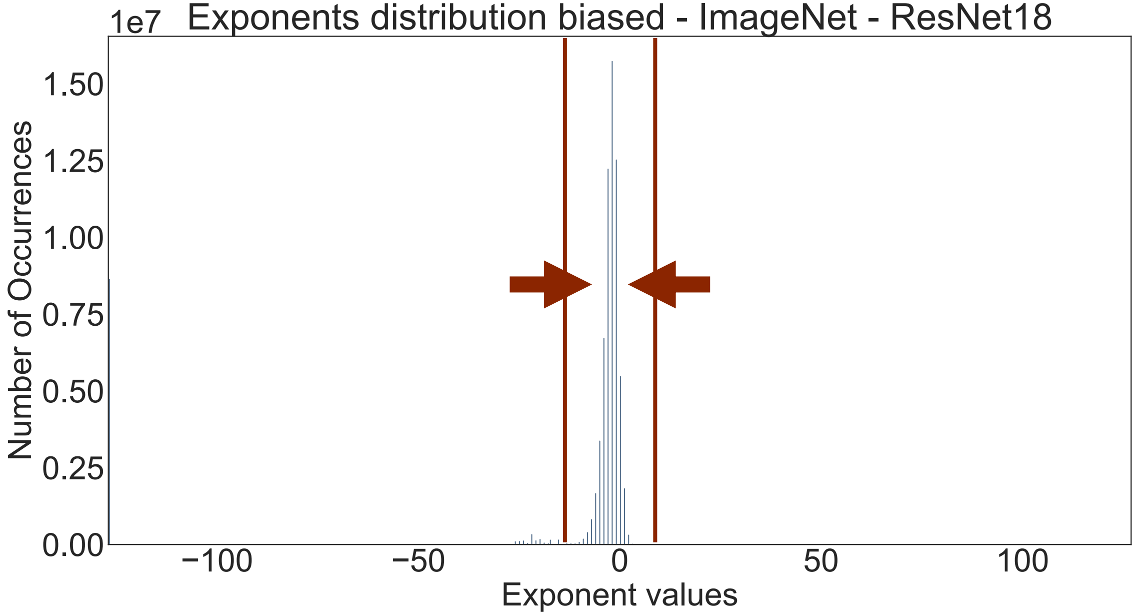
- Distribution
- Compress range
- $Value = (-1)^S \times M \times 2^E$

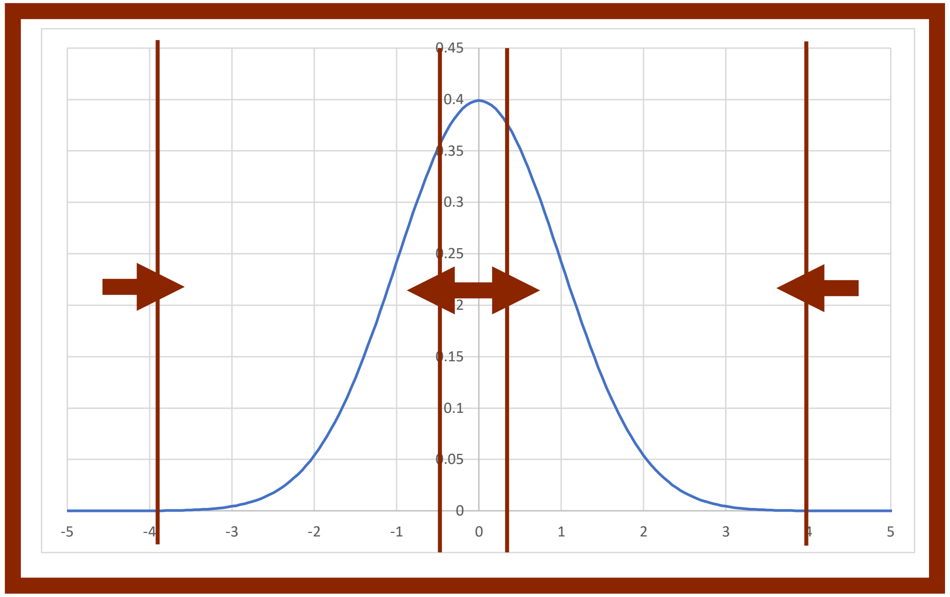




Quantum Exponent Range

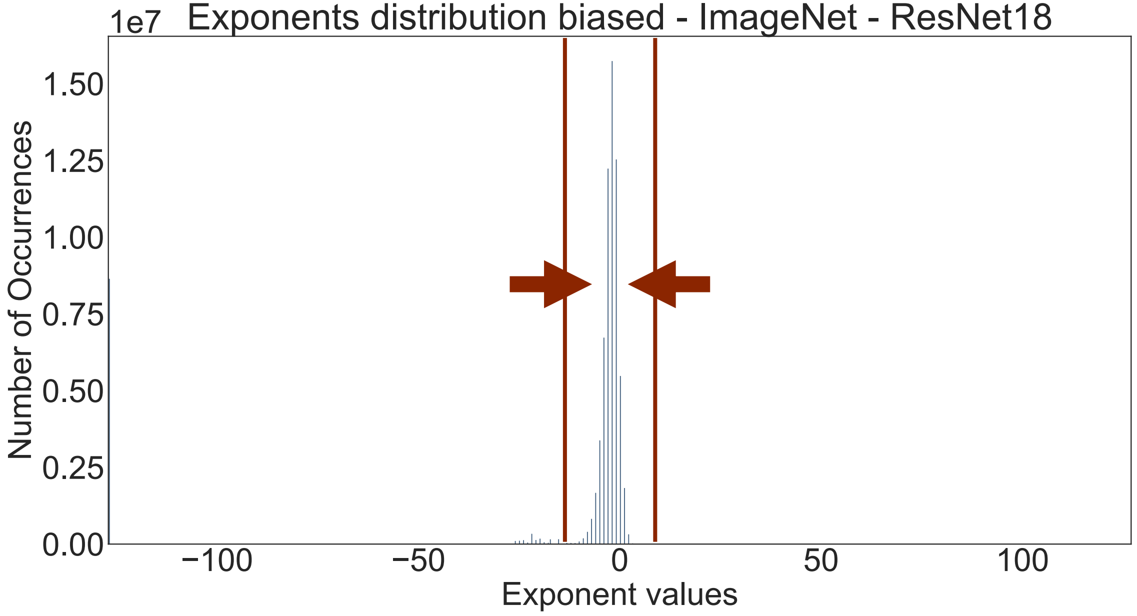
- Distribution
- Compress range
- $Value = (-1)^S \times M \times 2^E$

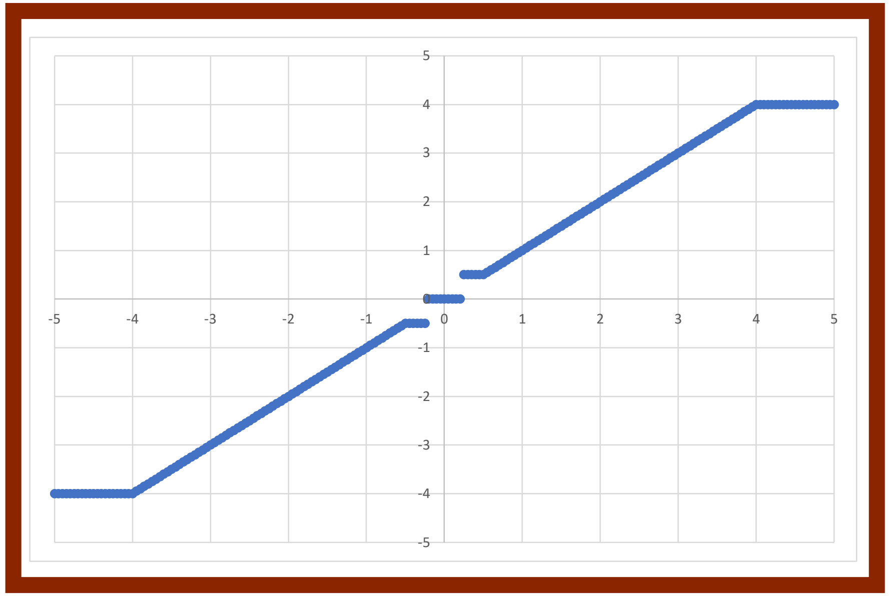


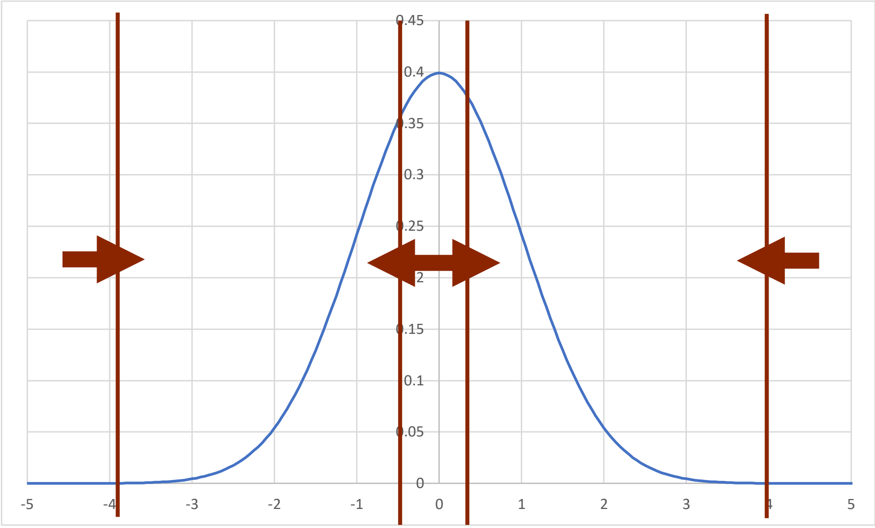


Quantum Exponent Range

- Distribution
- Compress range
- $Value = (-1)^{S} \times M \times 2^{E}$
- Range(x) overlay

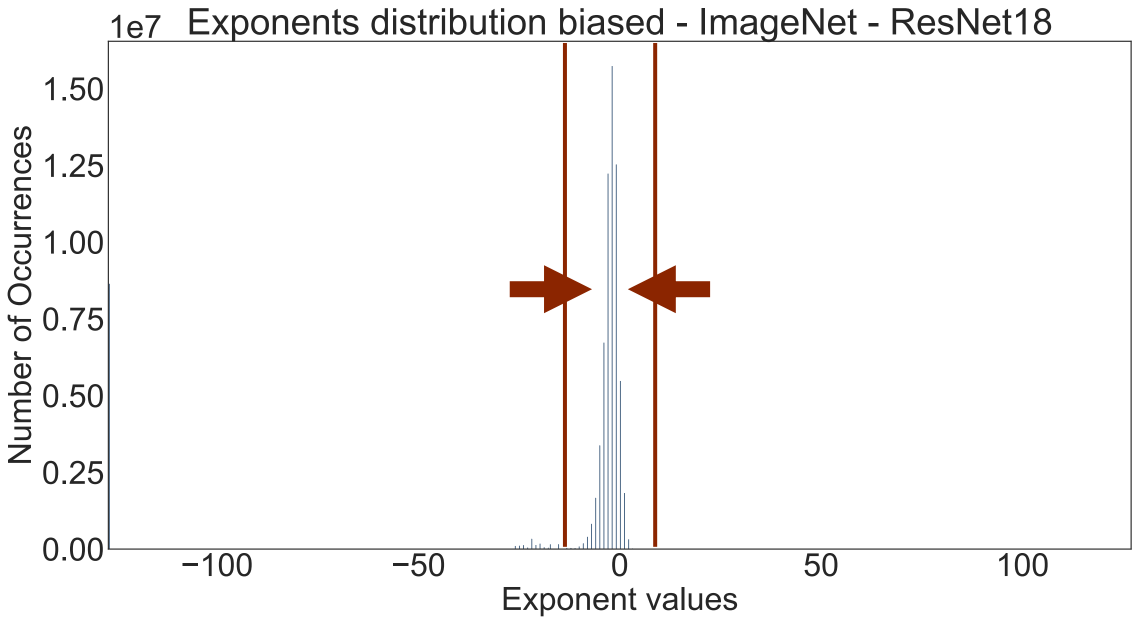


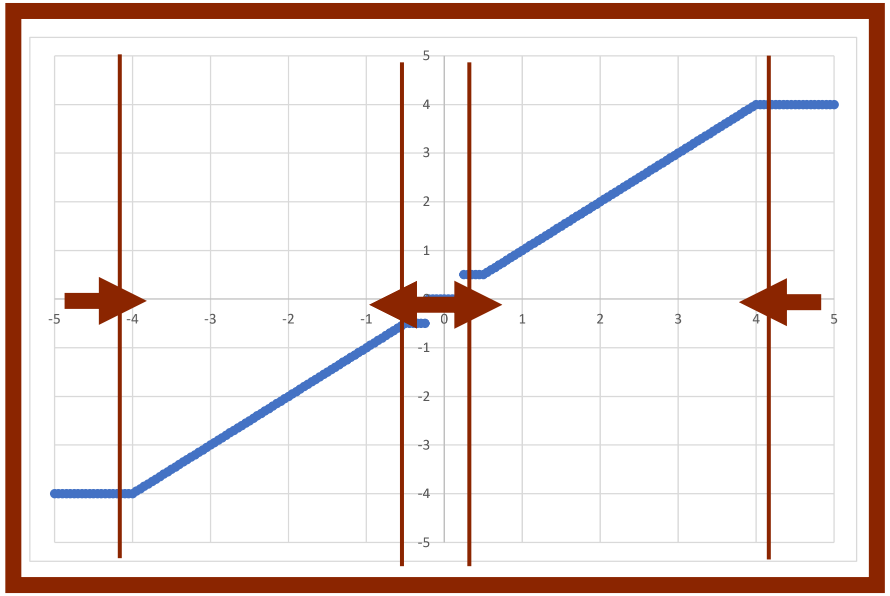


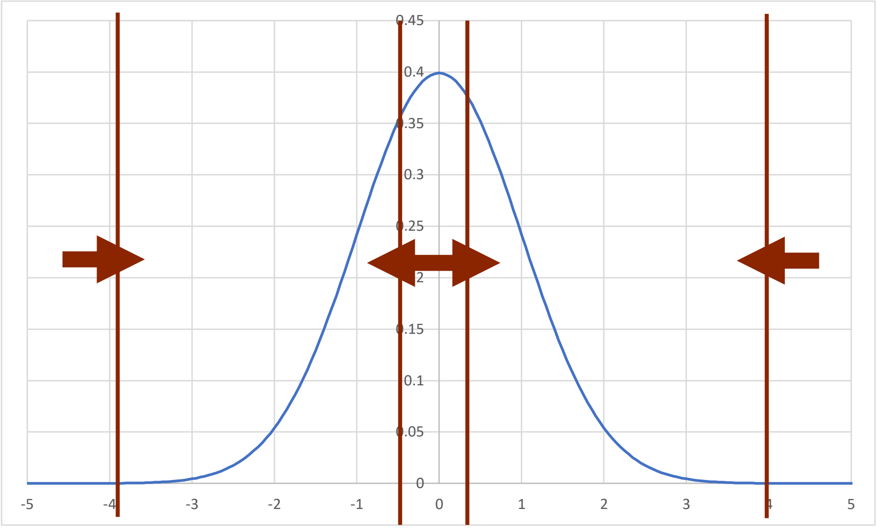


Quantum Exponent Range

- Distribution
- Compress range
- $Value = (-1)^S \times M \times 2^E$
- Range(x) overlay







Quantum Exponent

- Our Loss
    - $L = L_O + \gamma \times (\sum \lambda_i \times \alpha_i)$
    - Weighted according priority

Quantum Exponent

- Our Loss

  Original Loss

  - $L = L_O + \gamma \times (\sum \lambda_i \times \alpha_i)$
  - Weighted according priority

Quantum Exponent

- Our Loss

  Exponent Lengths

  - $L = L_O + \gamma \times (\sum \lambda_i \times \alpha_i)$
  - Weighted according priority

Quantum Exponent

- Our Loss
  - $L = L_O + \gamma \times (\sum \lambda_i \times \alpha_i)$
  - Weighted according priority

Exponent Weights

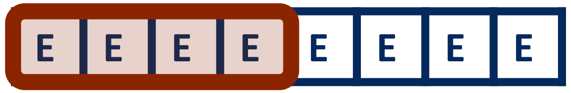
Quantum Exponent

- Our Loss
  - $L = L_O + \gamma \times (\sum \lambda_i \times \alpha_i)$
  - Weighted according priority

Regularizer Strength

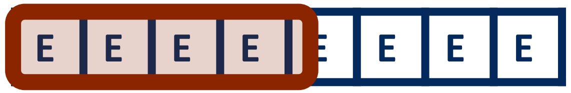
Quantum Exponent

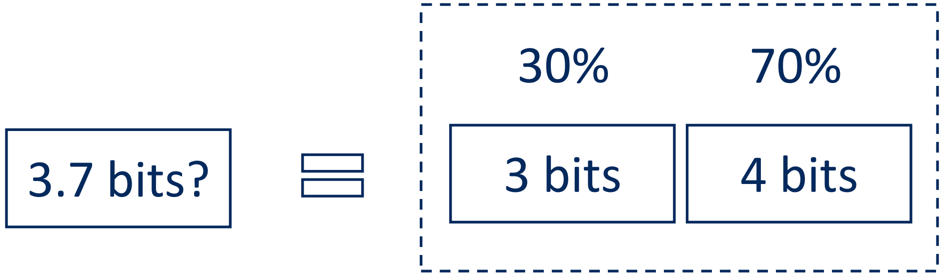
- Our Loss
  - $L = L_O + \gamma \times (\sum \lambda_i \times \alpha_i)$
  - Weighted according priority
- Int Datatype



Quantum Exponent

- Our Loss
  - $L = L_O + \gamma \times (\sum \lambda_i \times \alpha_i)$
  - Weighted according priority
- Int Datatype
- Non-Int Datatype





3.7 bits?

30%

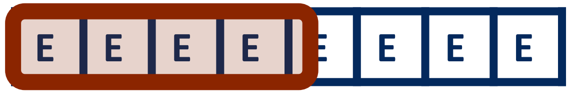
70%

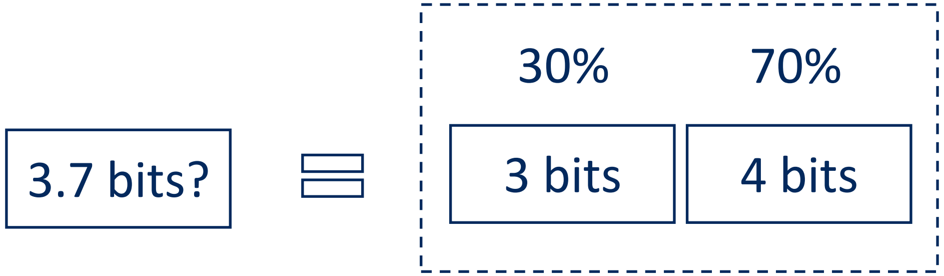
3 bits

4 bits

Quantum Exponent

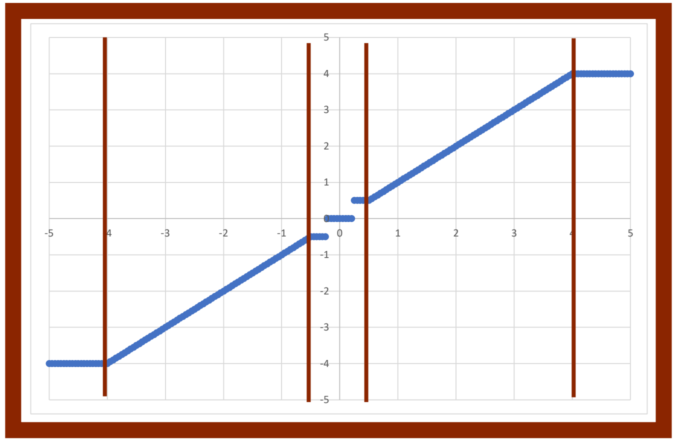
- Our Loss
  - $L = L_O + \gamma \times (\sum \lambda_i \times \alpha_i)$
  - Weighted according priority
- Int Datatype
- Non-Int Datatype
- Determine boundaries




3.7 bits?

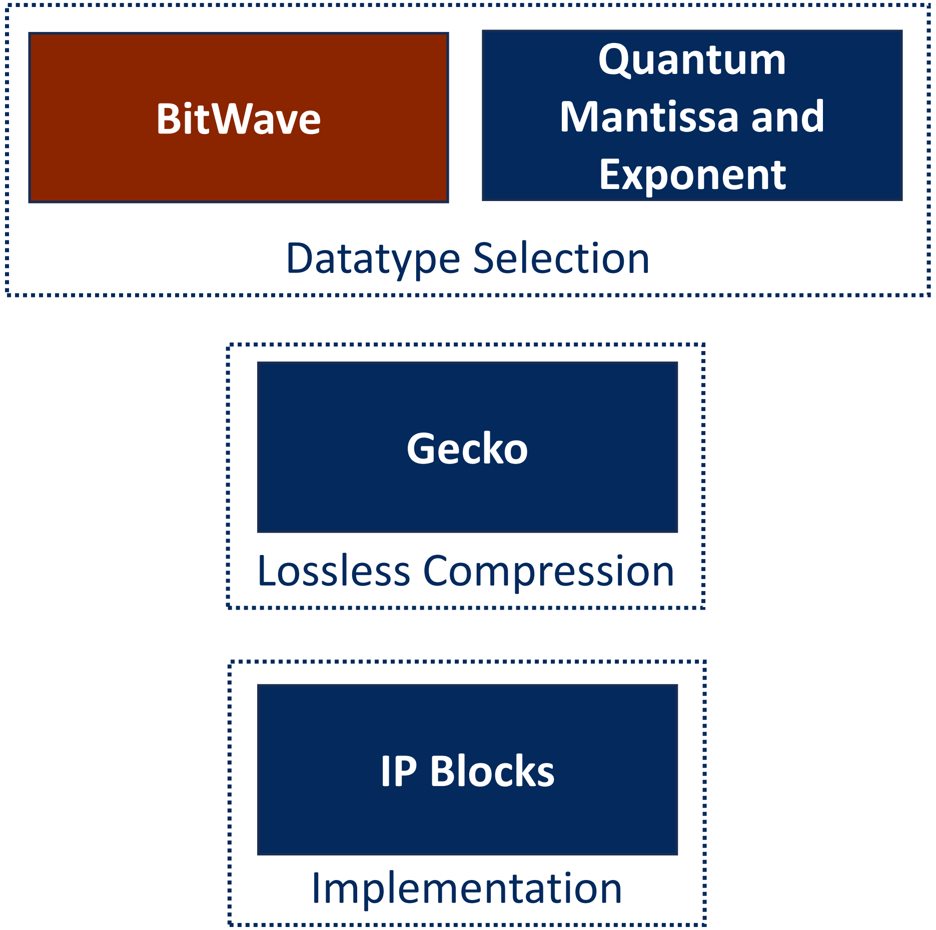
30% 70%

3 bits 4 bits



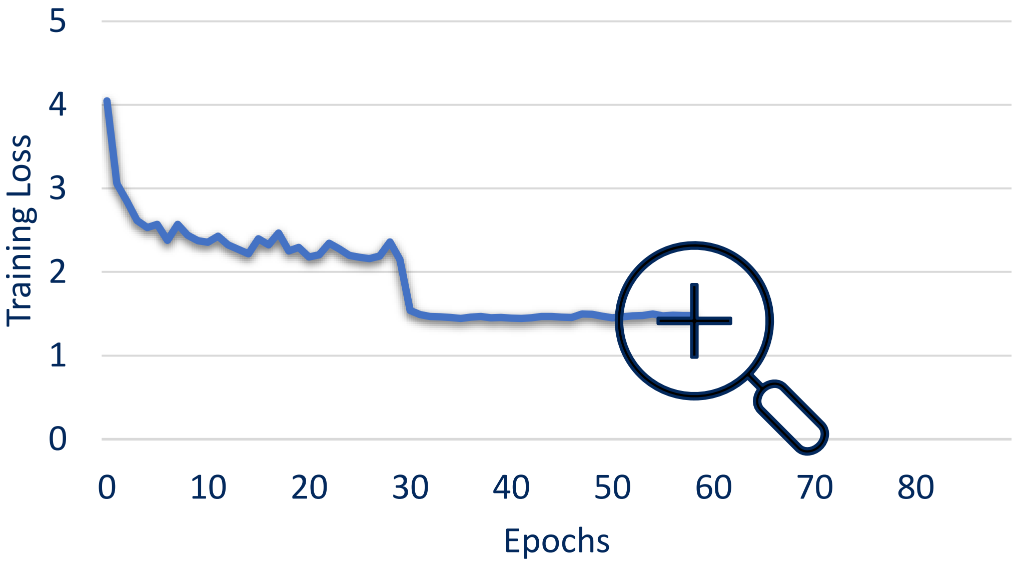
Schrödinger’s FP

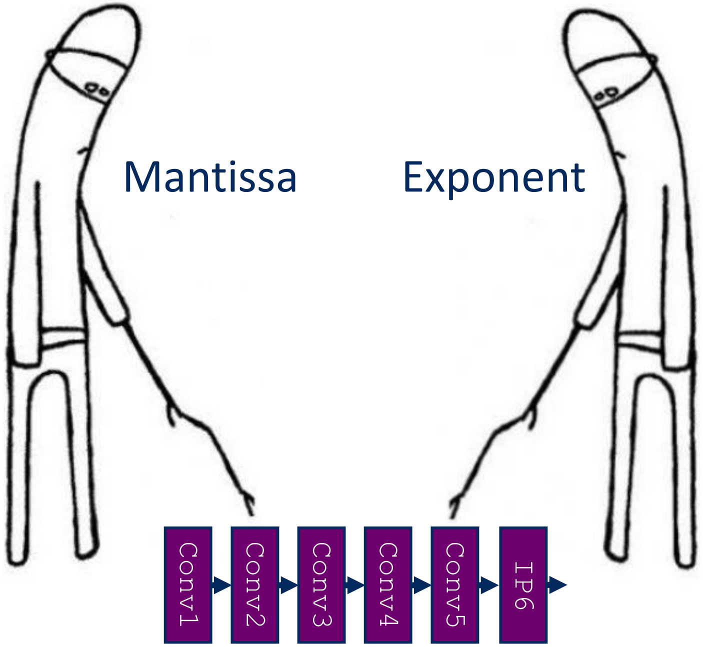
- Machine Learning
  - *Quantum Mantissa and Exponent*
- Black Box Sampling
  - *BitWave*
- Exponent compression
  - *Gecko*
- Hardware IP blocks



BitWave

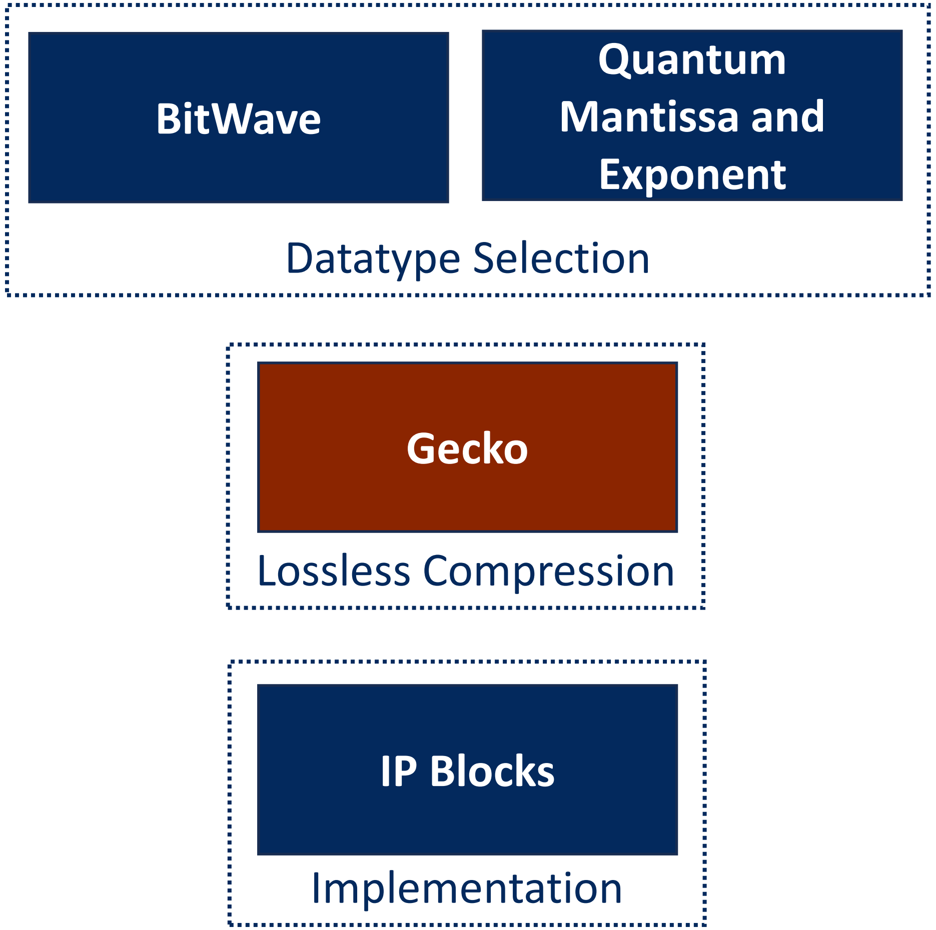
- Observe Loss
  - Adjust bitlengths
    - Mantissa
    - Exponent
- Training is forgiving
- Network wide





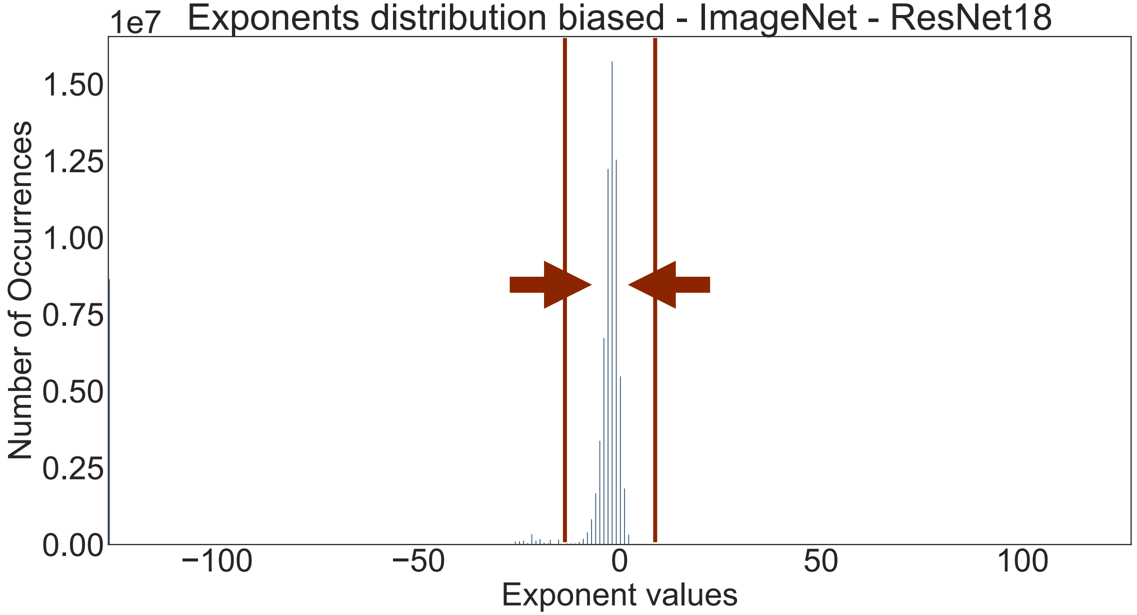
Schrödinger's FP

- Machine Learning
  - *Quantum Mantissa and Exponent*
- Black Box Sampling
  - *BitWave*
- Exponent compression
  - *Gecko*
- Hardware IP blocks



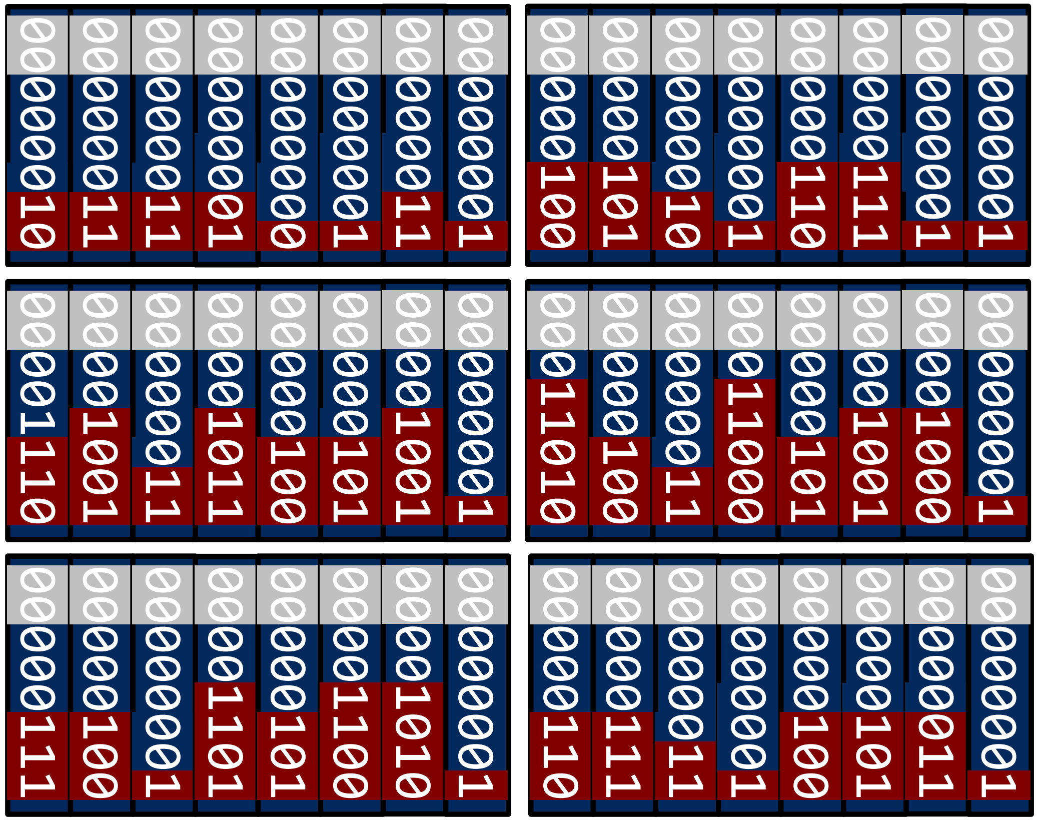
Exponent – *Gecko*

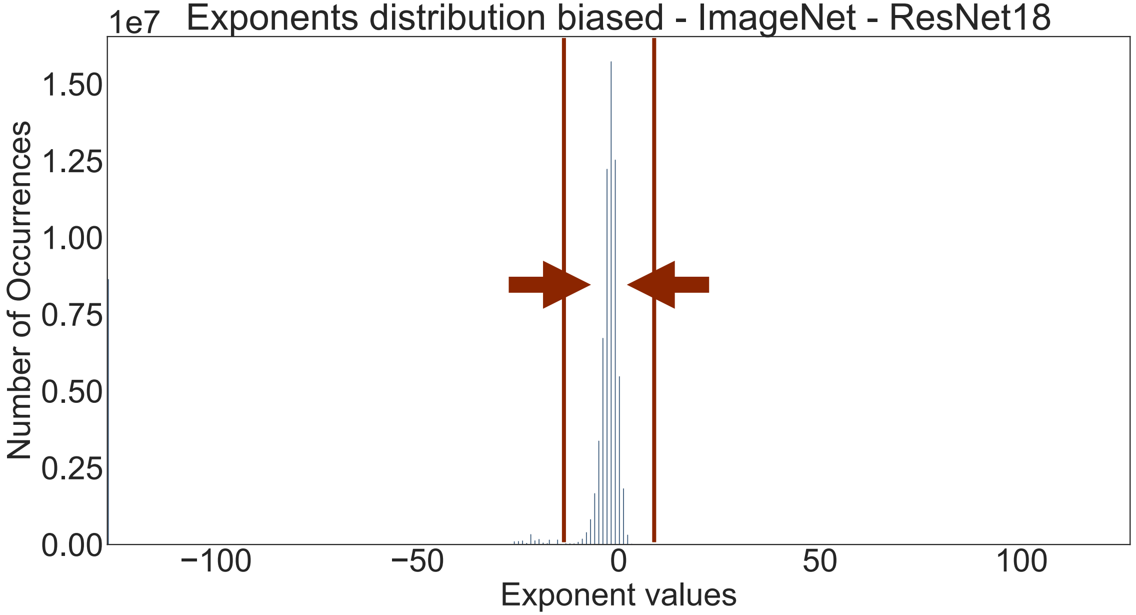
- Distribution



Exponent – *Gecko*

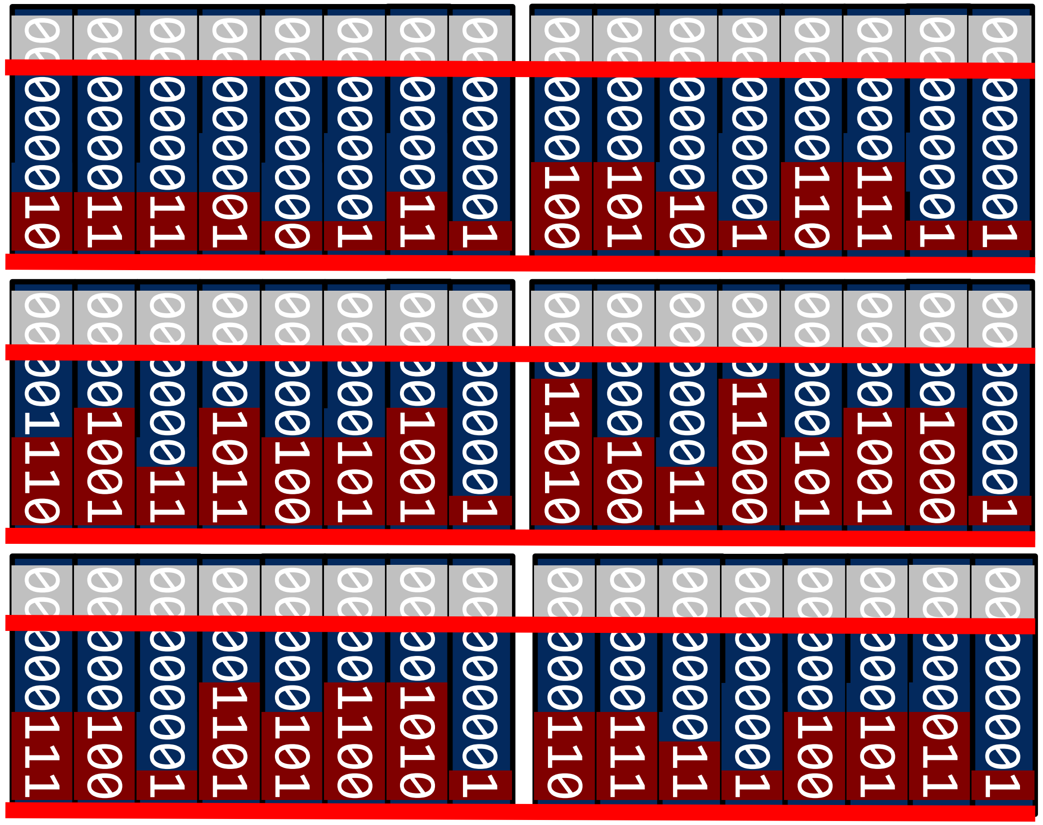
- Distribution
- Group

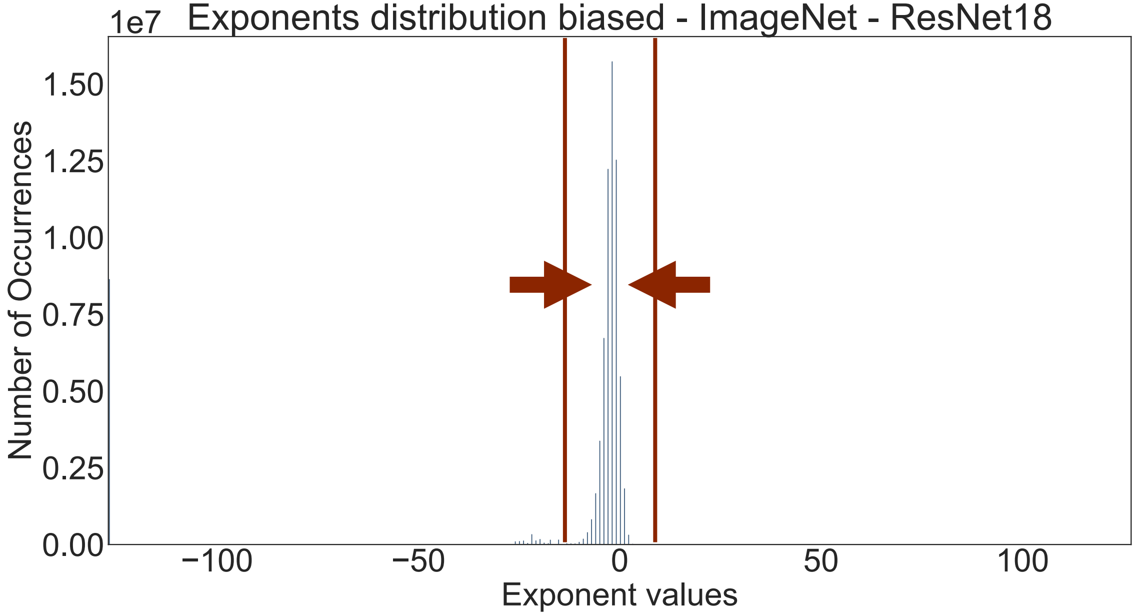




Exponent – *Gecko*

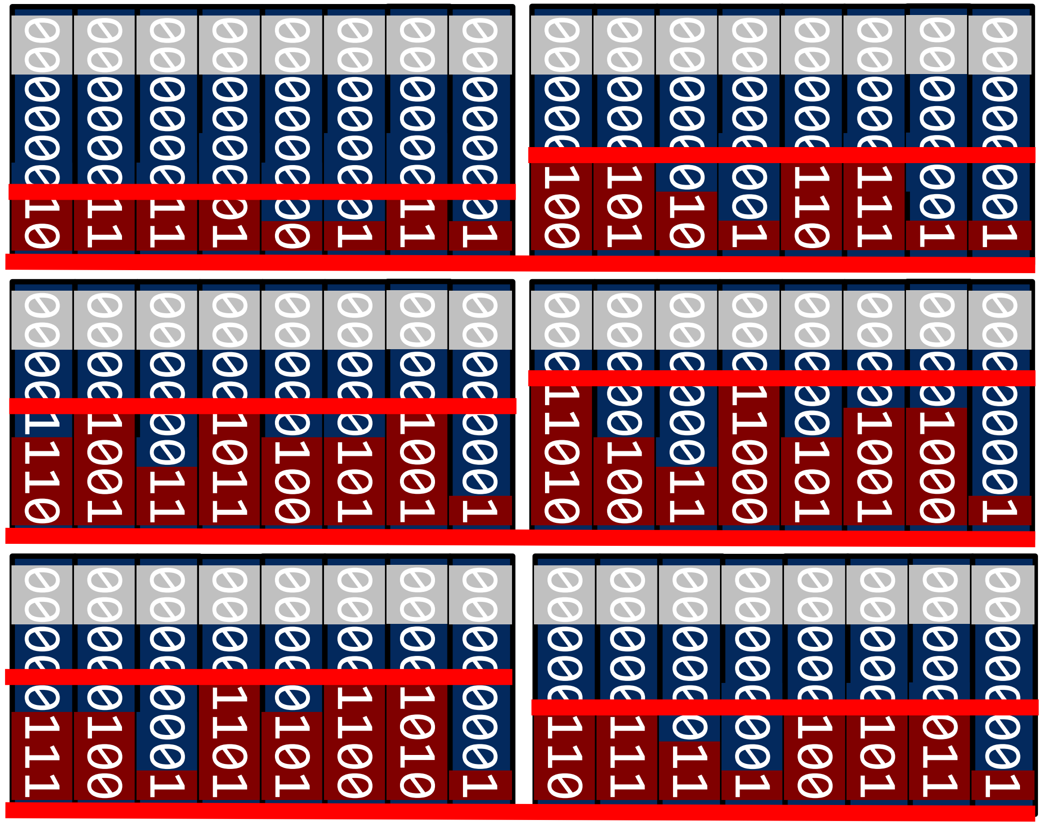
- Distribution
- Group

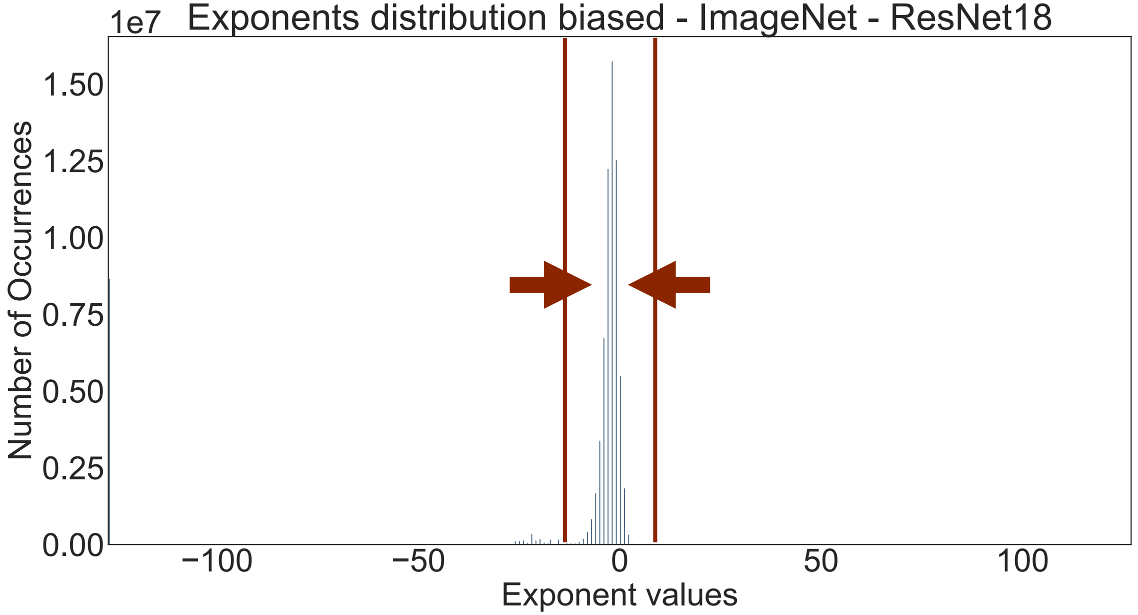




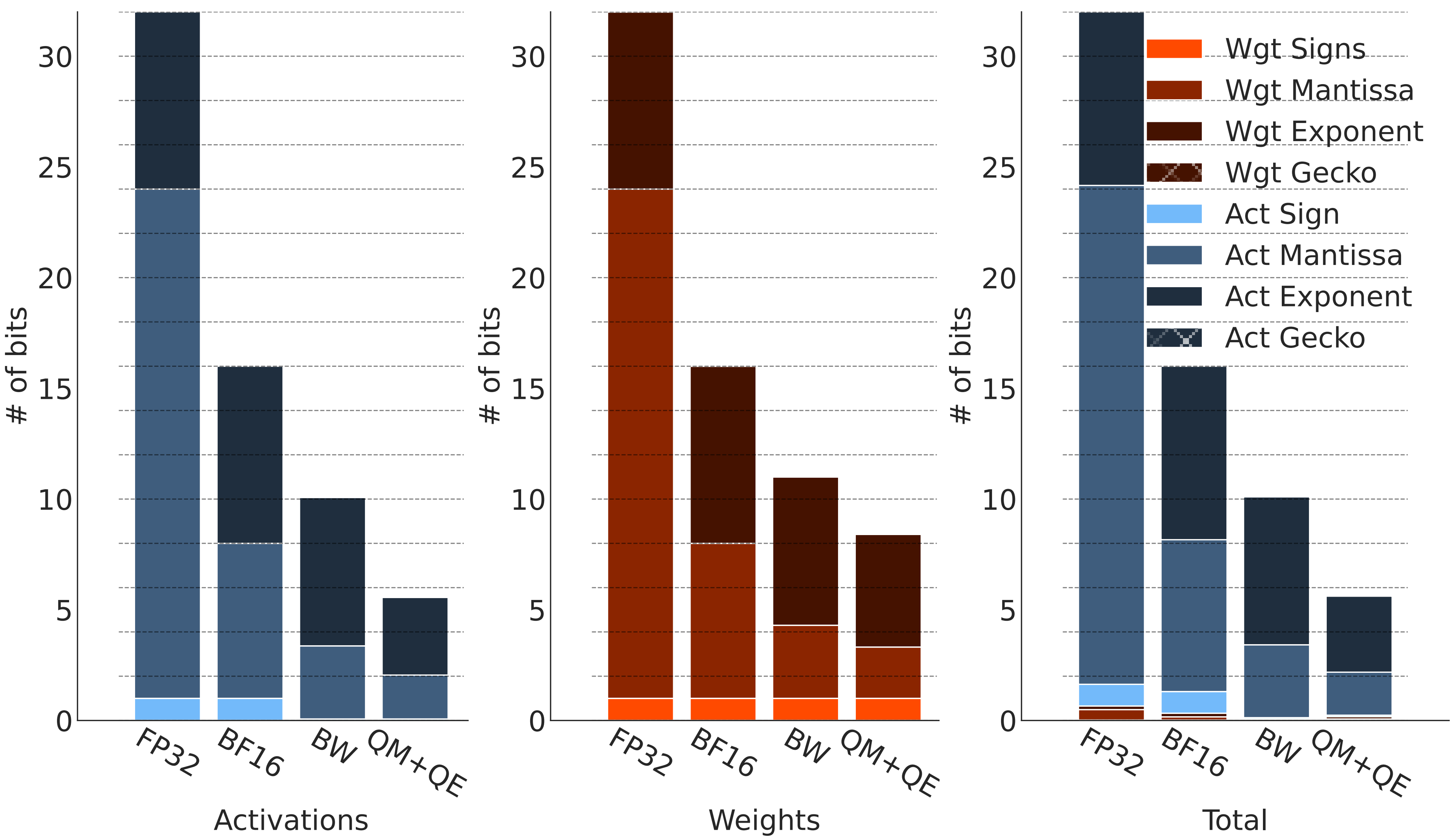
Exponent – *Gecko*

- Distribution
- Group

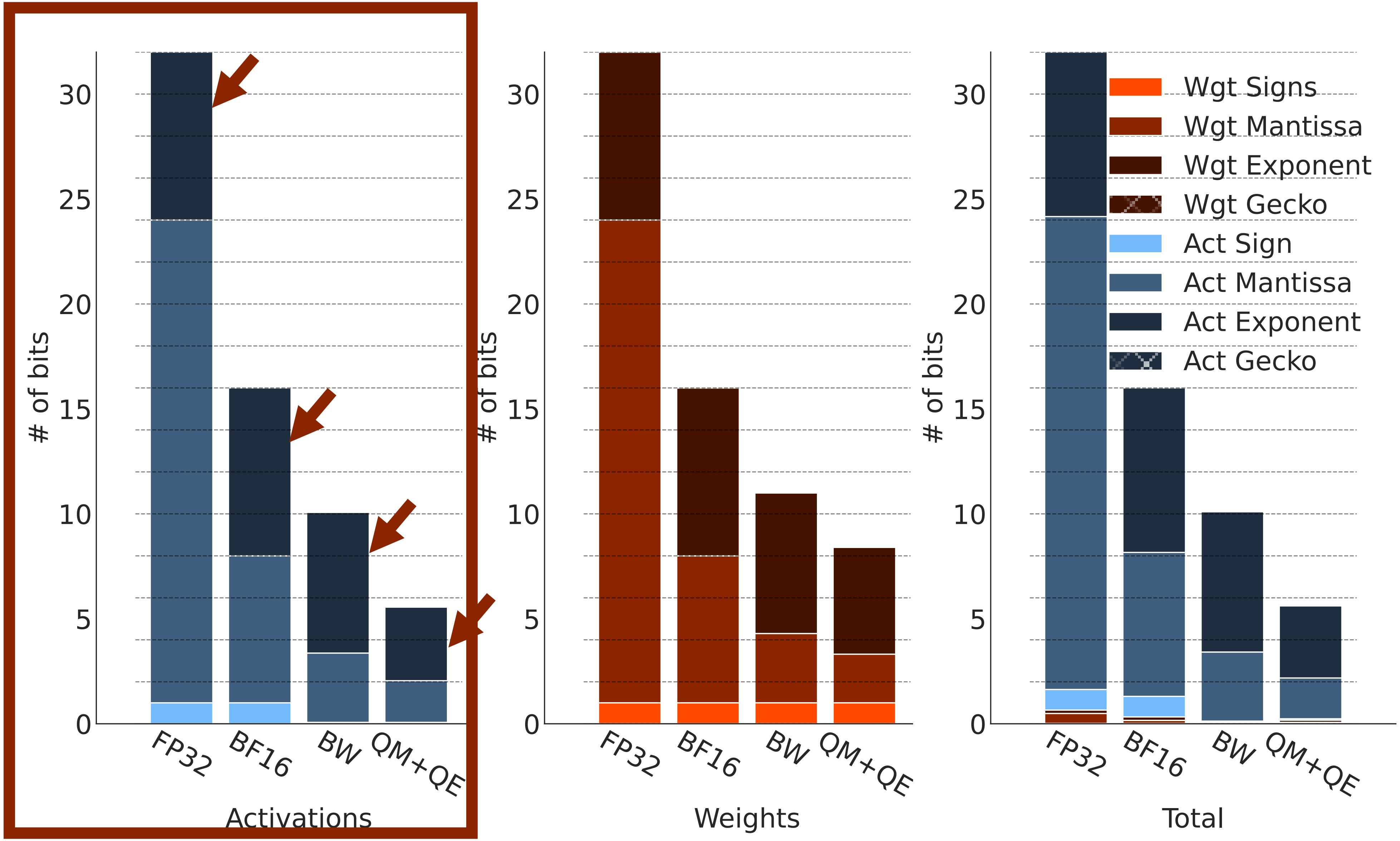




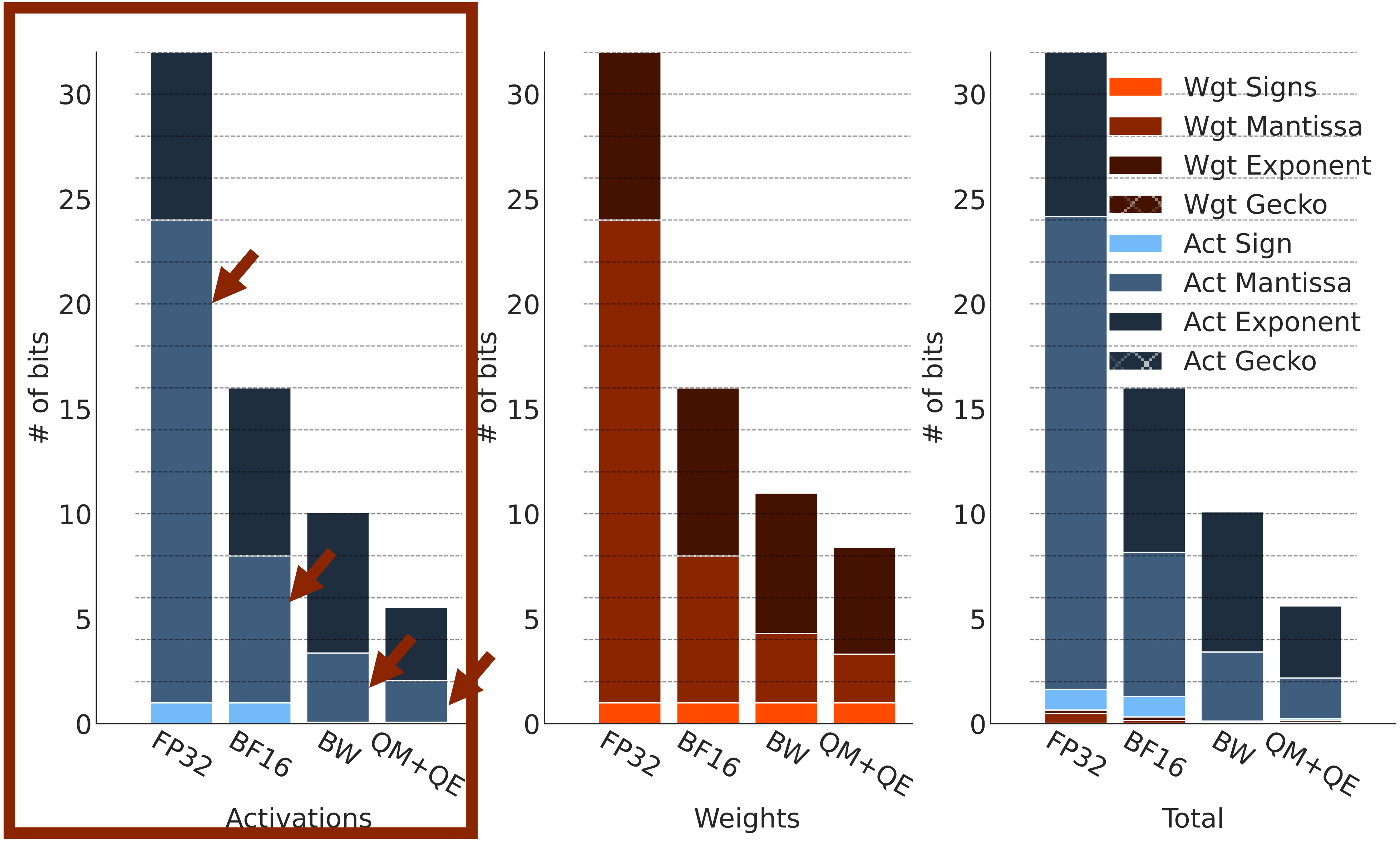
Massive footprint reduction on ResNet18!



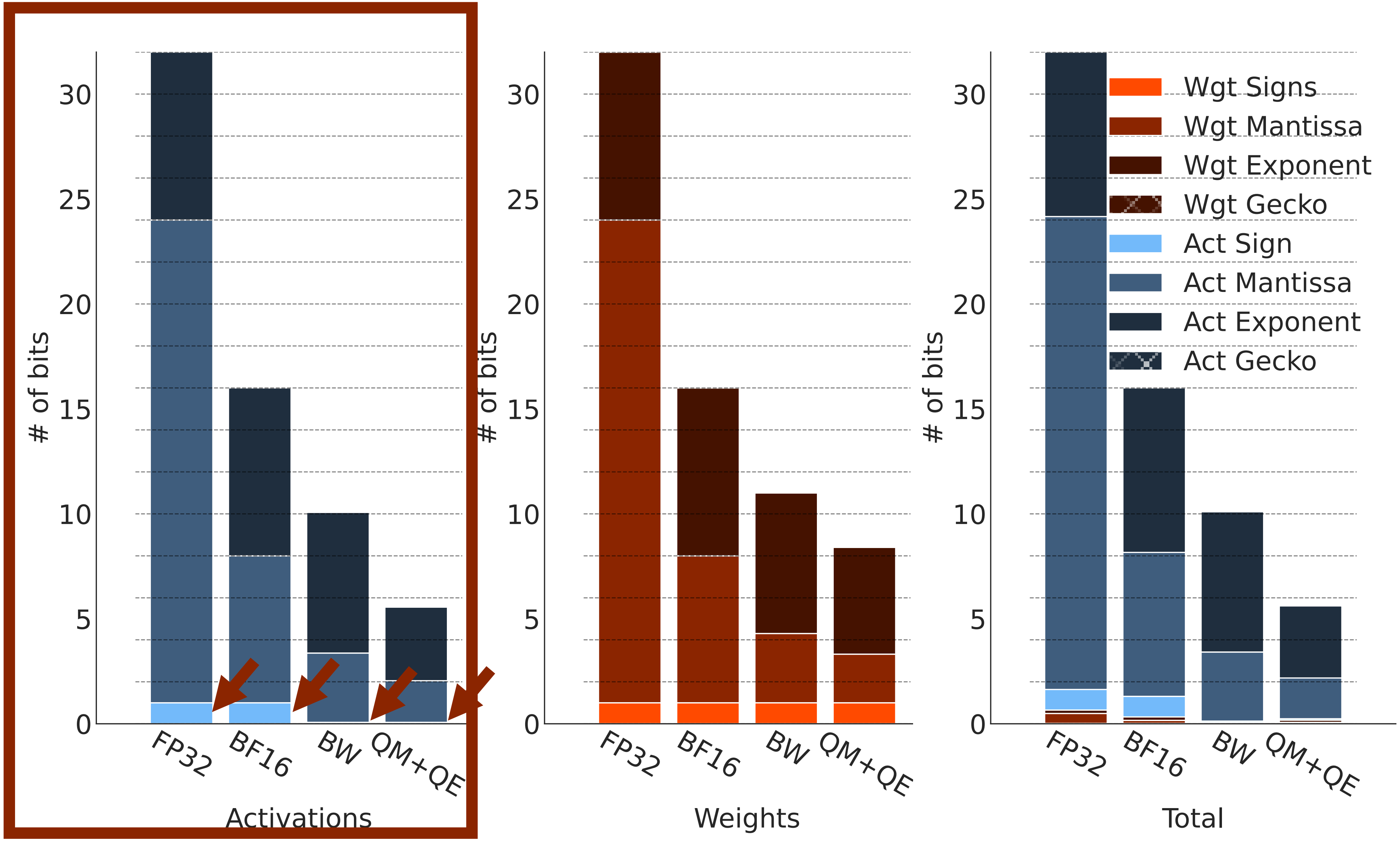
Massive footprint reduction on ResNet18!



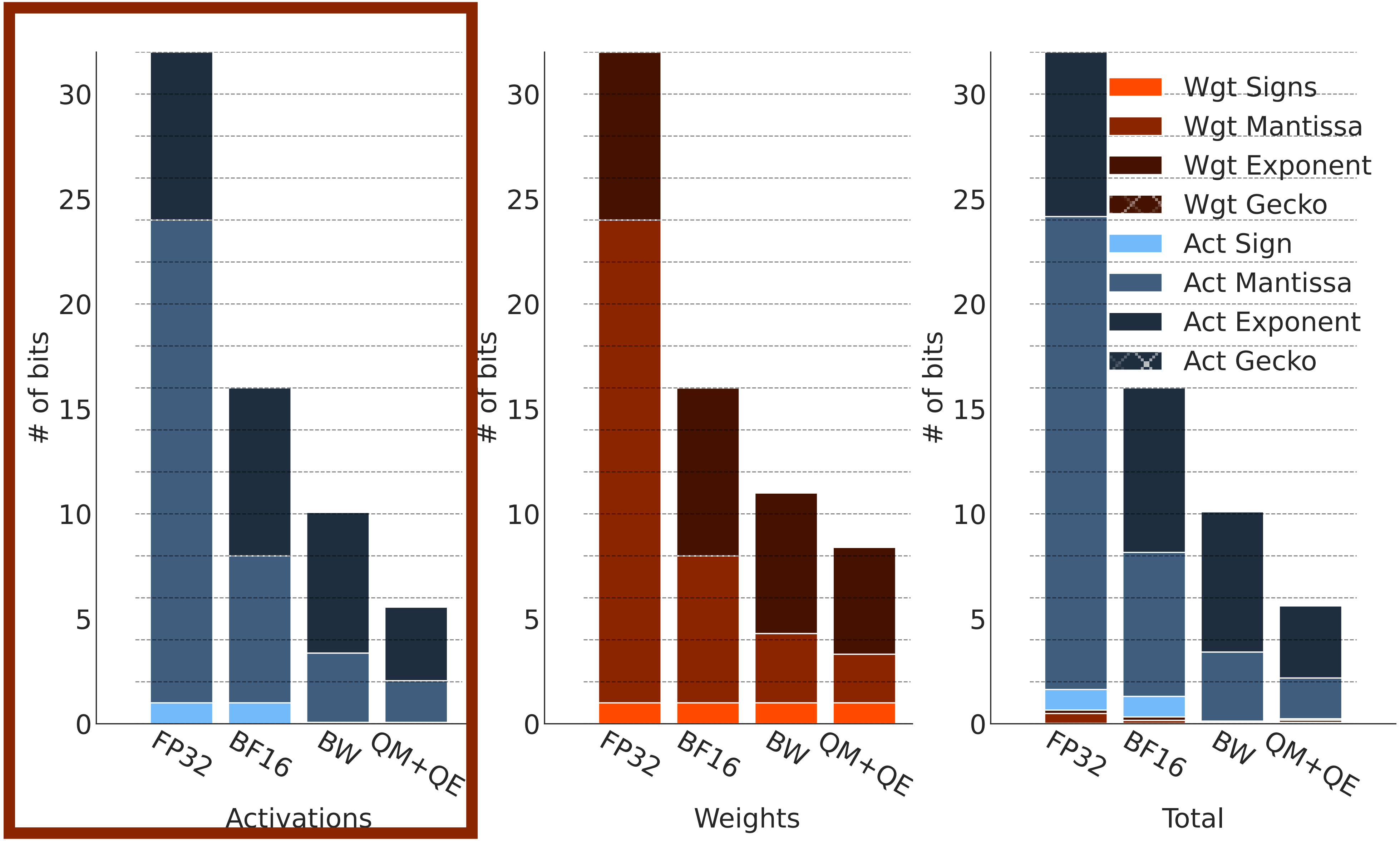
Massive footprint reduction on ResNet18!



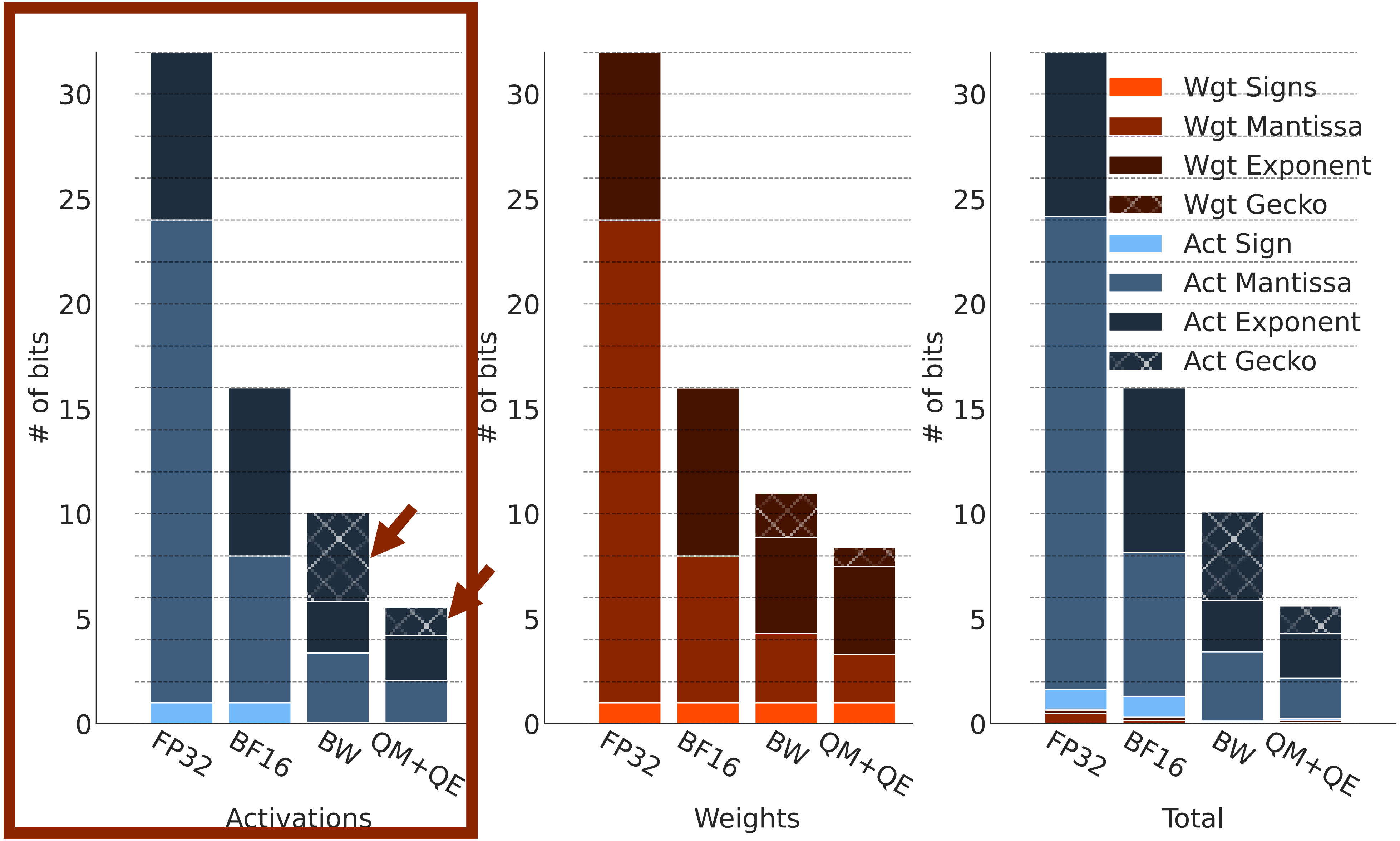
Massive footprint reduction on ResNet18!



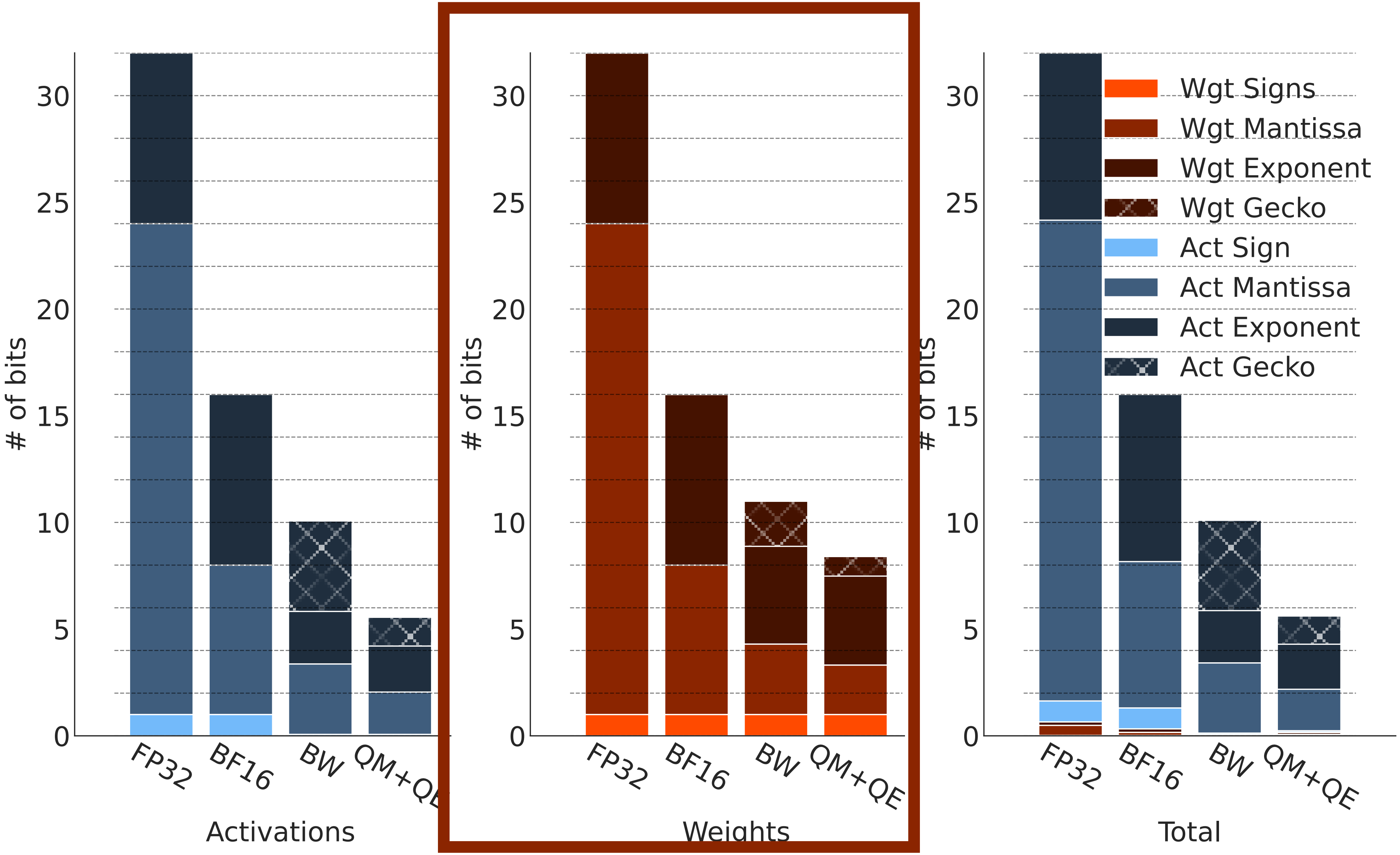
Massive footprint reduction on ResNet18!



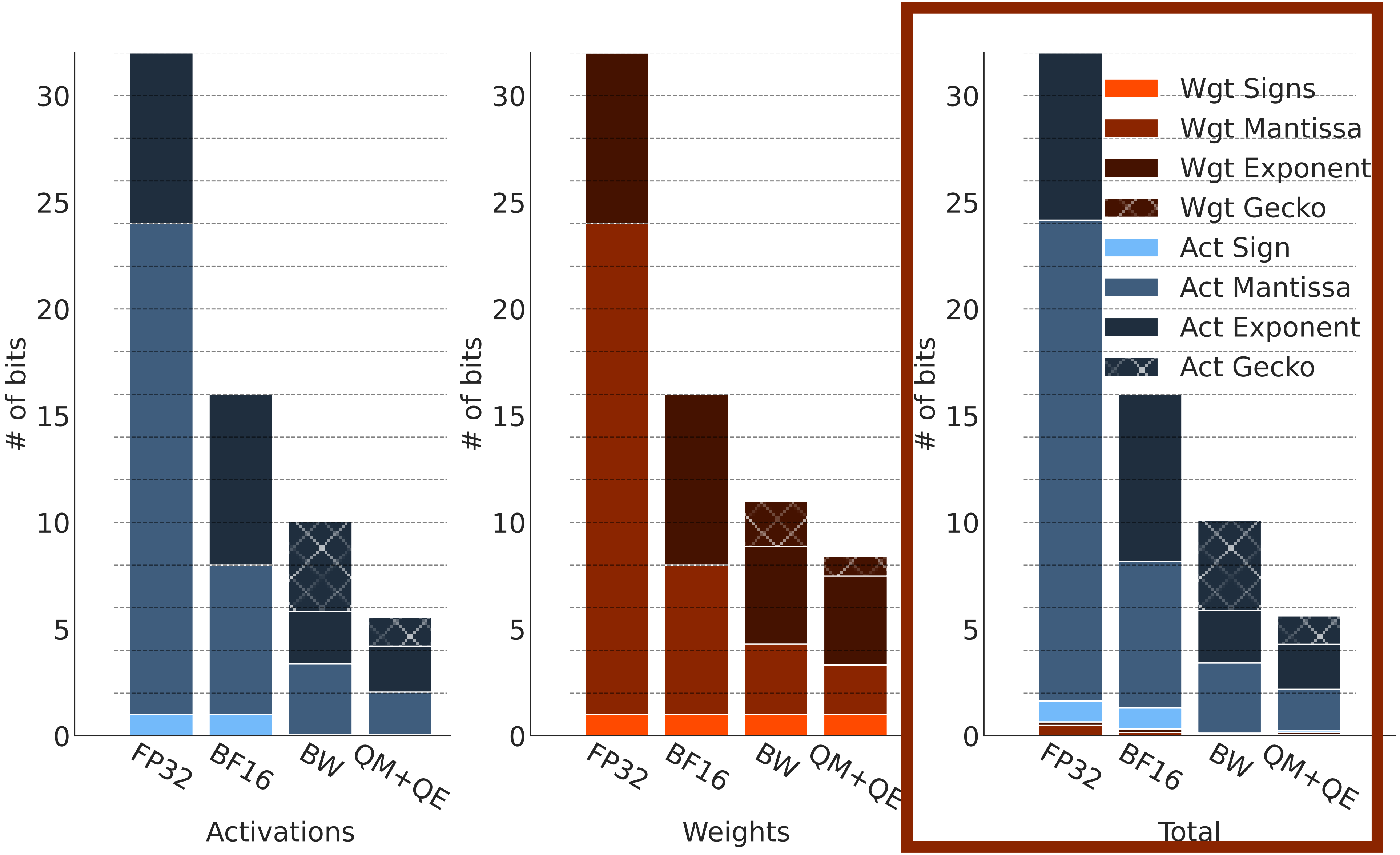
Massive footprint reduction on ResNet18!



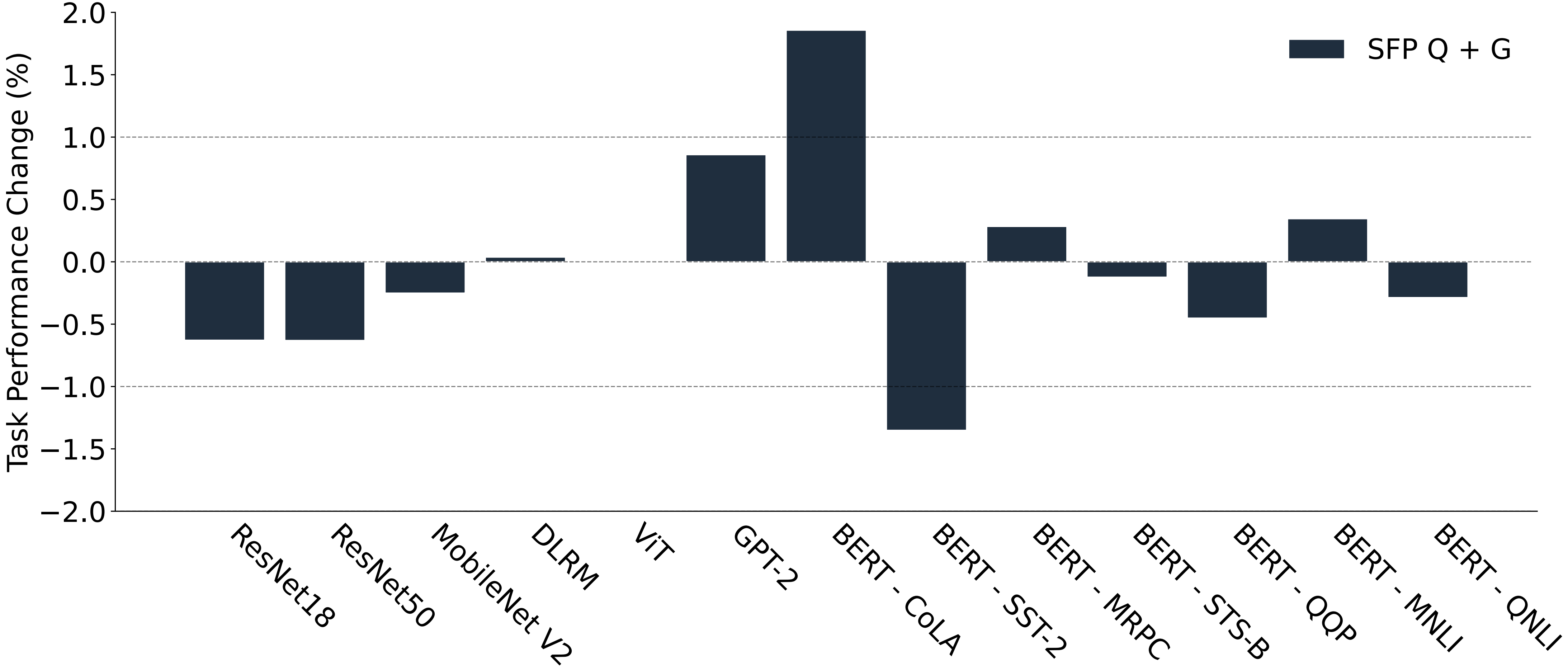
Massive footprint reduction on ResNet18!



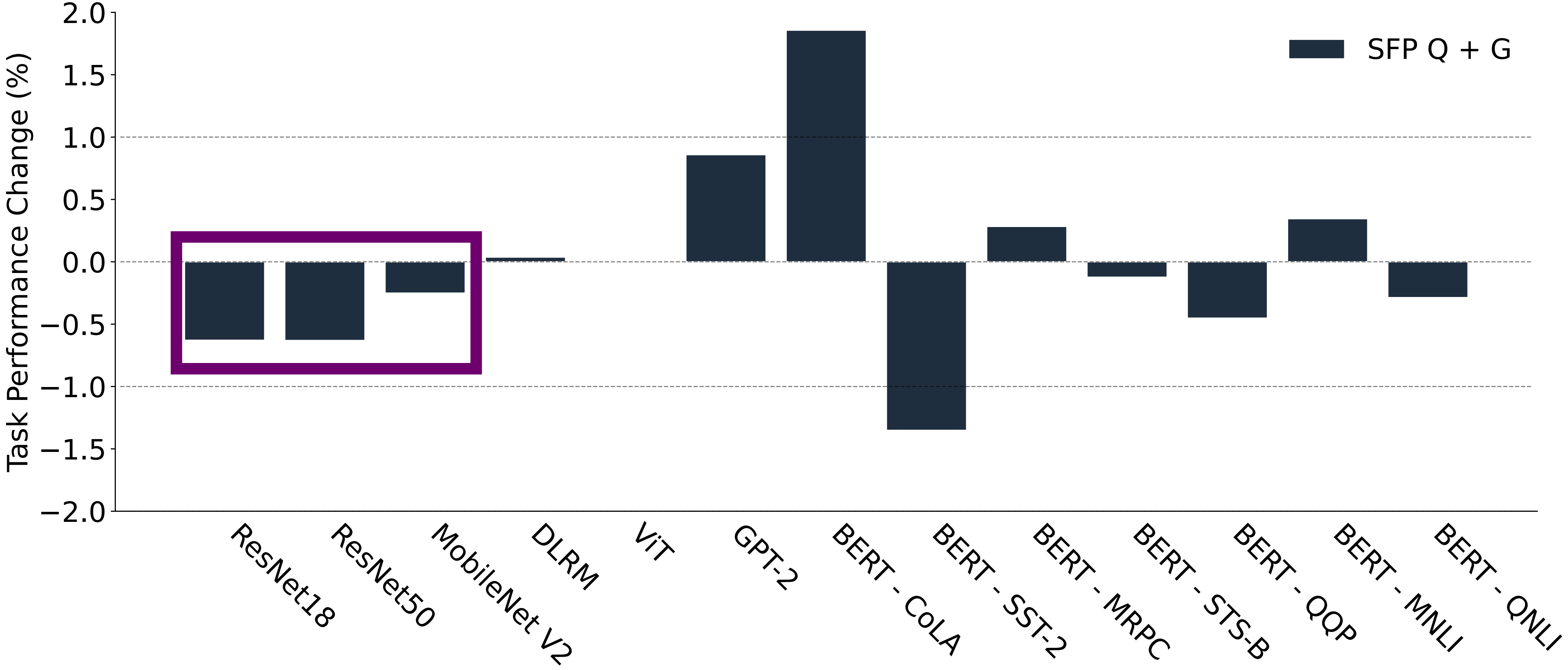
Massive footprint reduction on ResNet18!



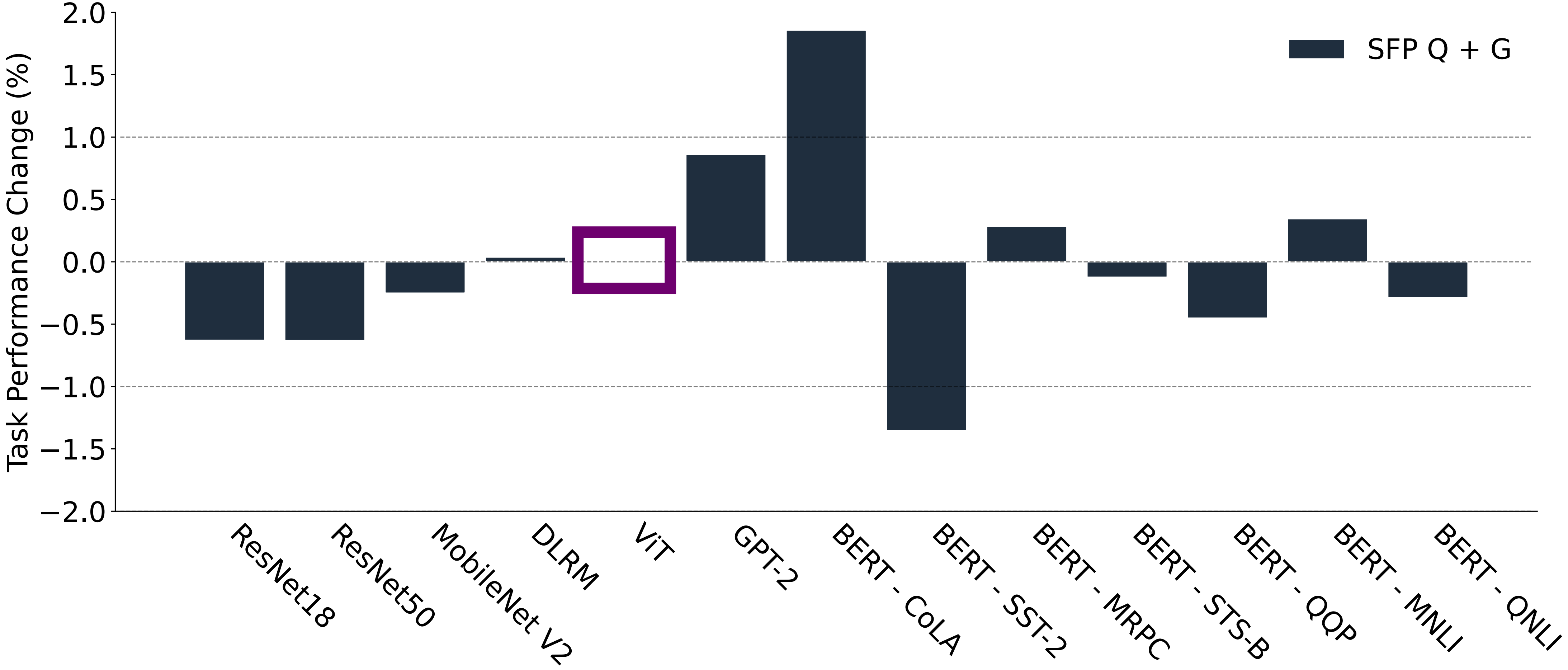
Similar Task Performance!



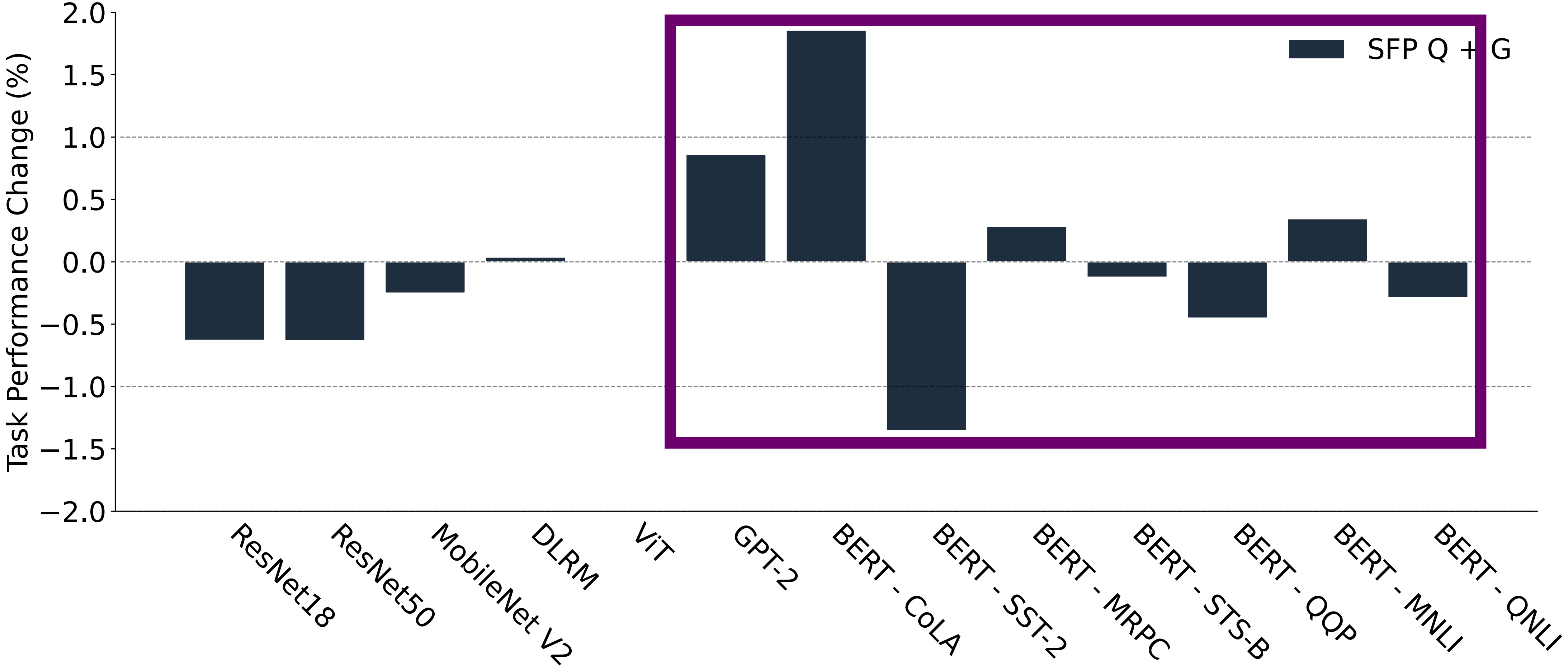
Similar Task Performance!



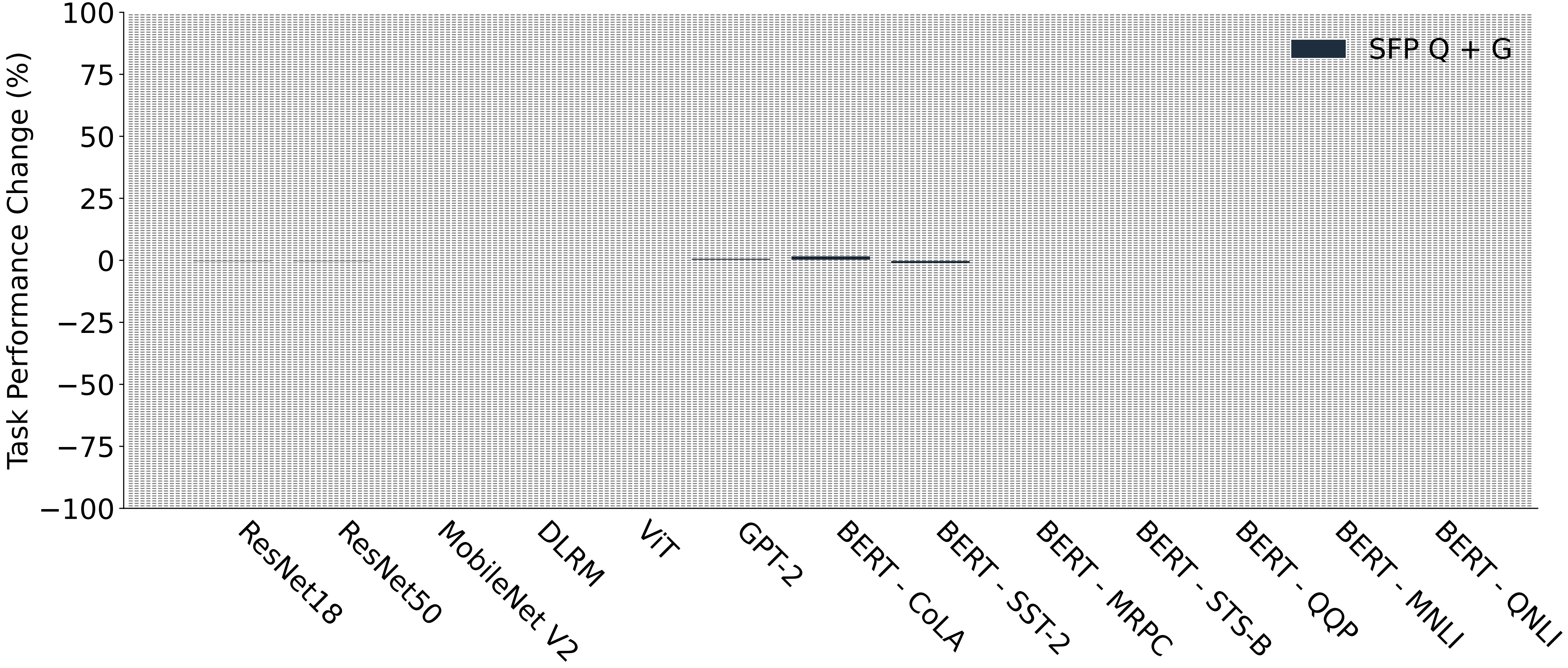
Similar Task Performance!



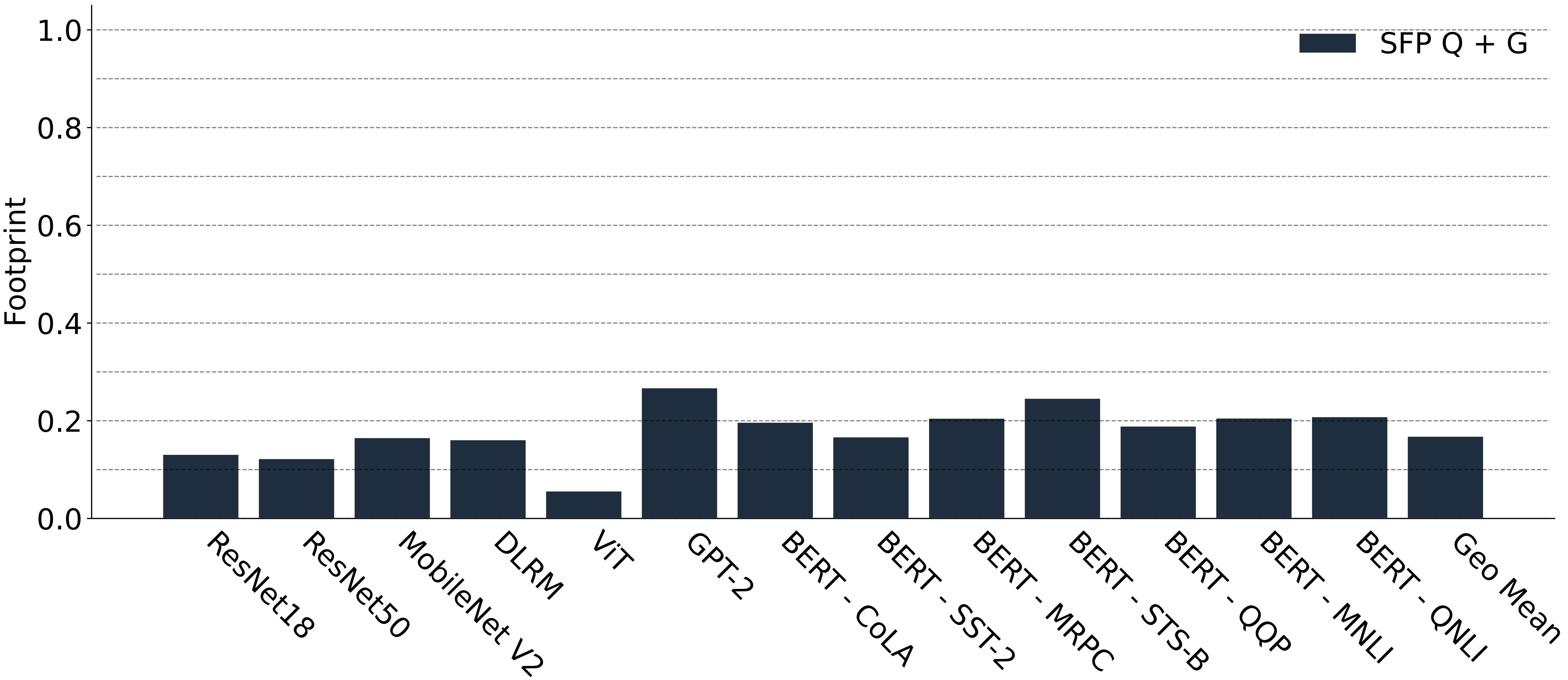
Similar Task Performance!



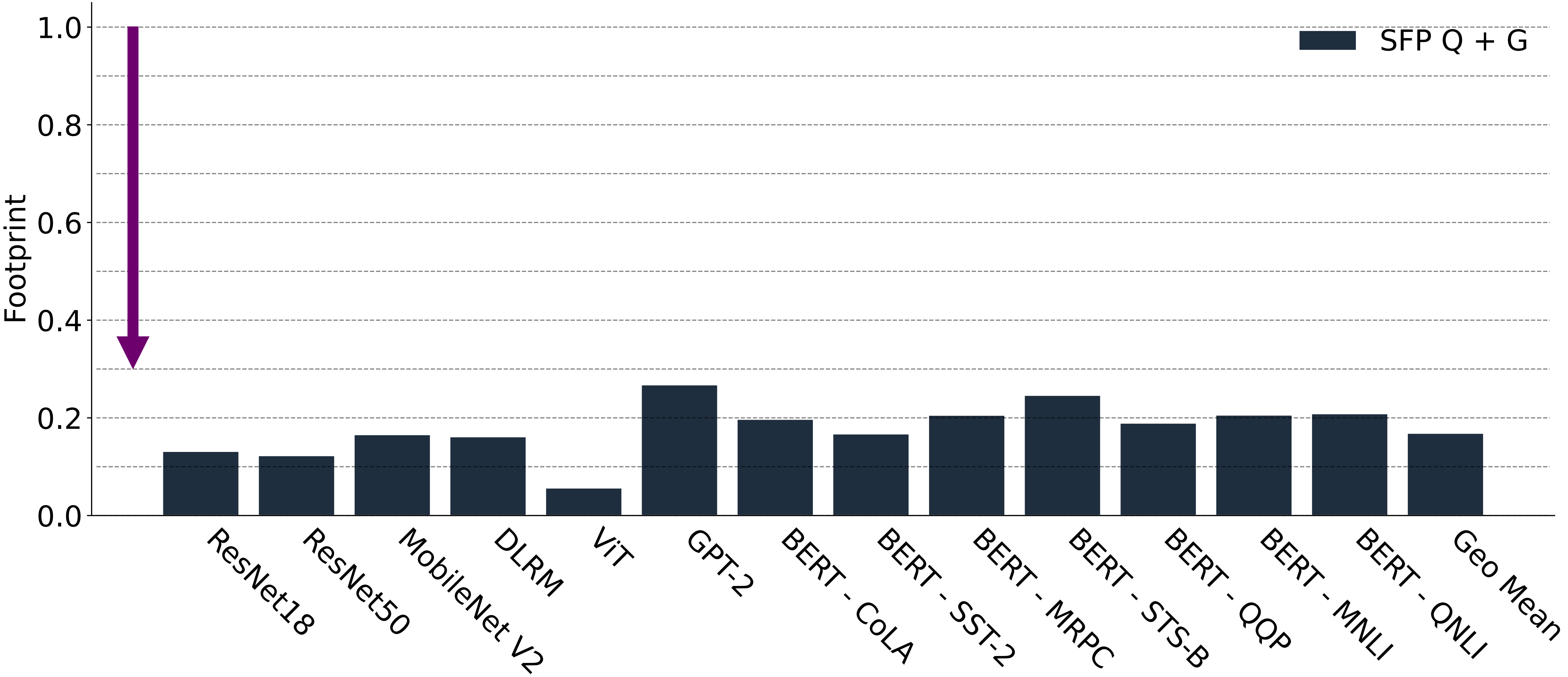
Similar Task Performance!



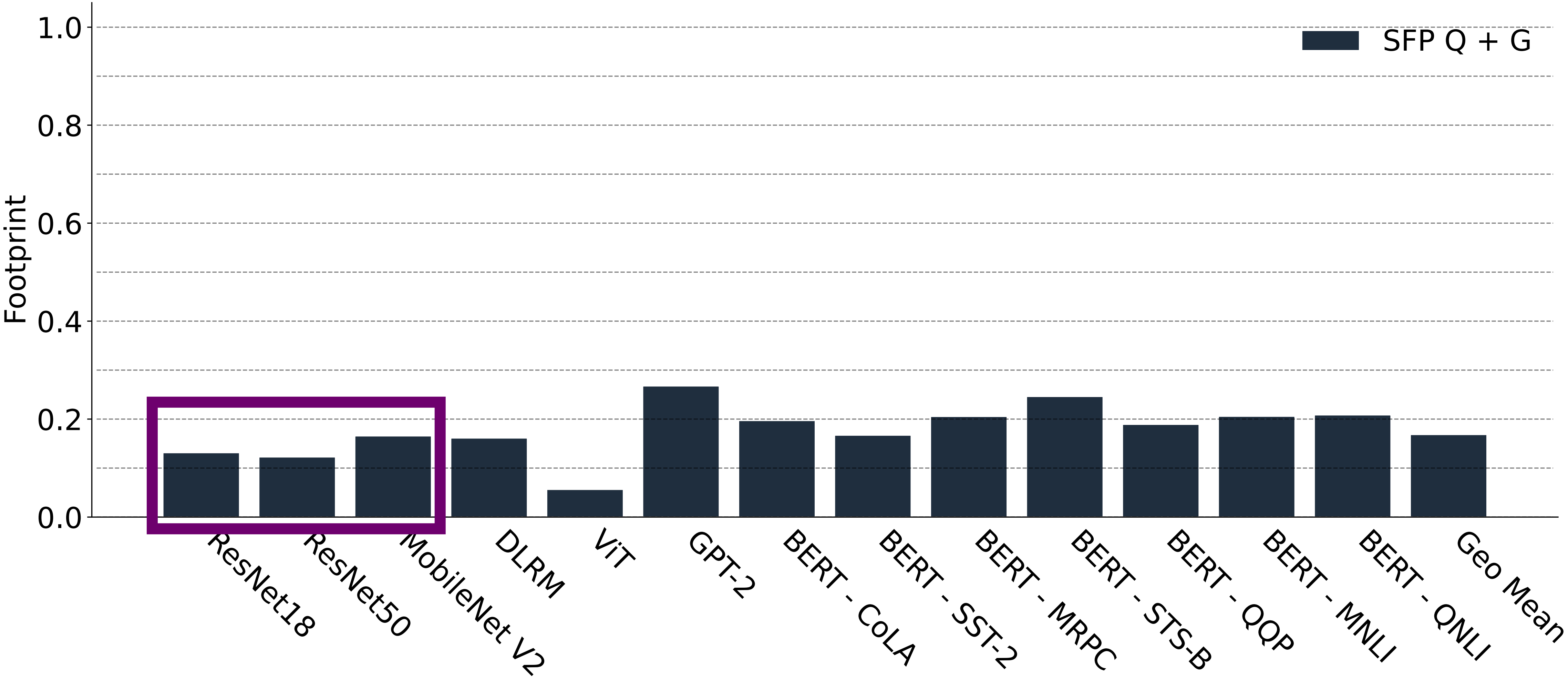
Massive footprint reduction across the board!



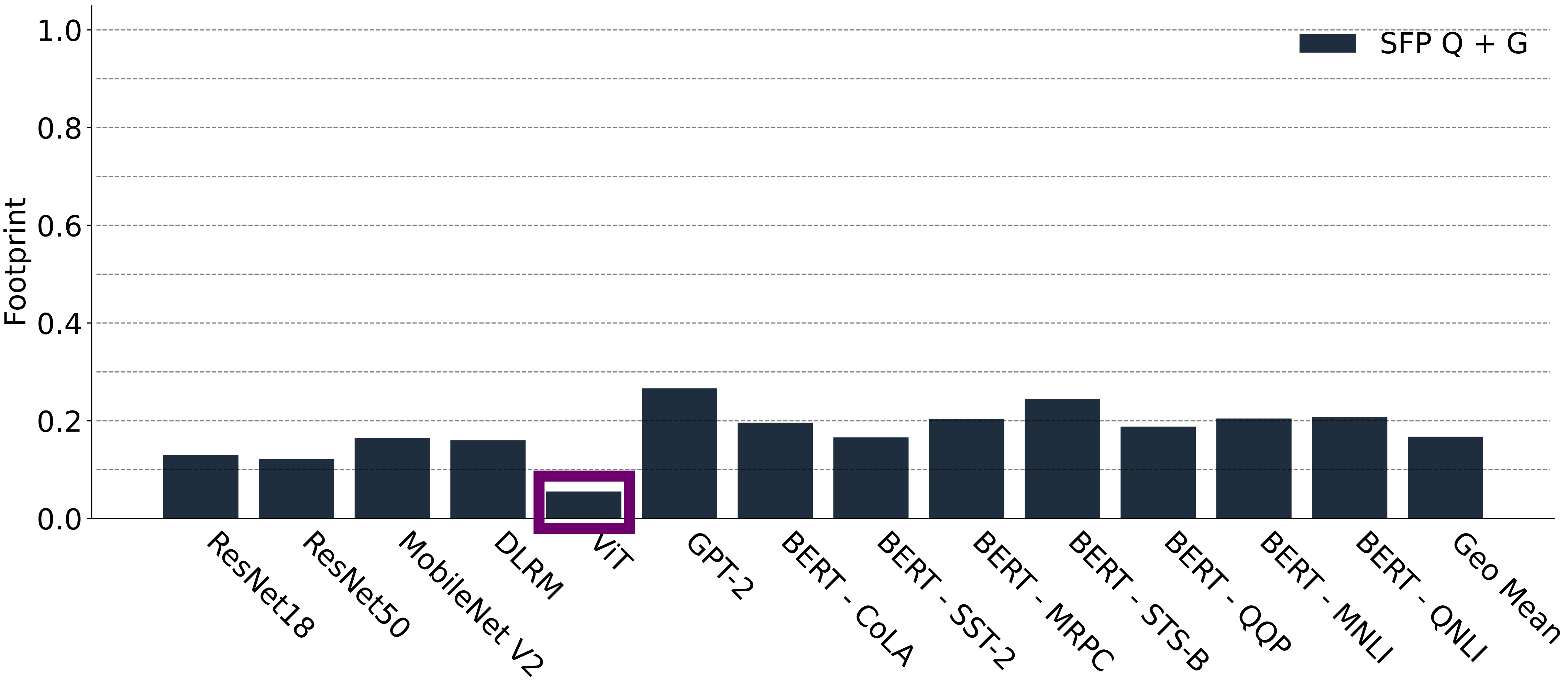
Massive footprint reduction across the board!



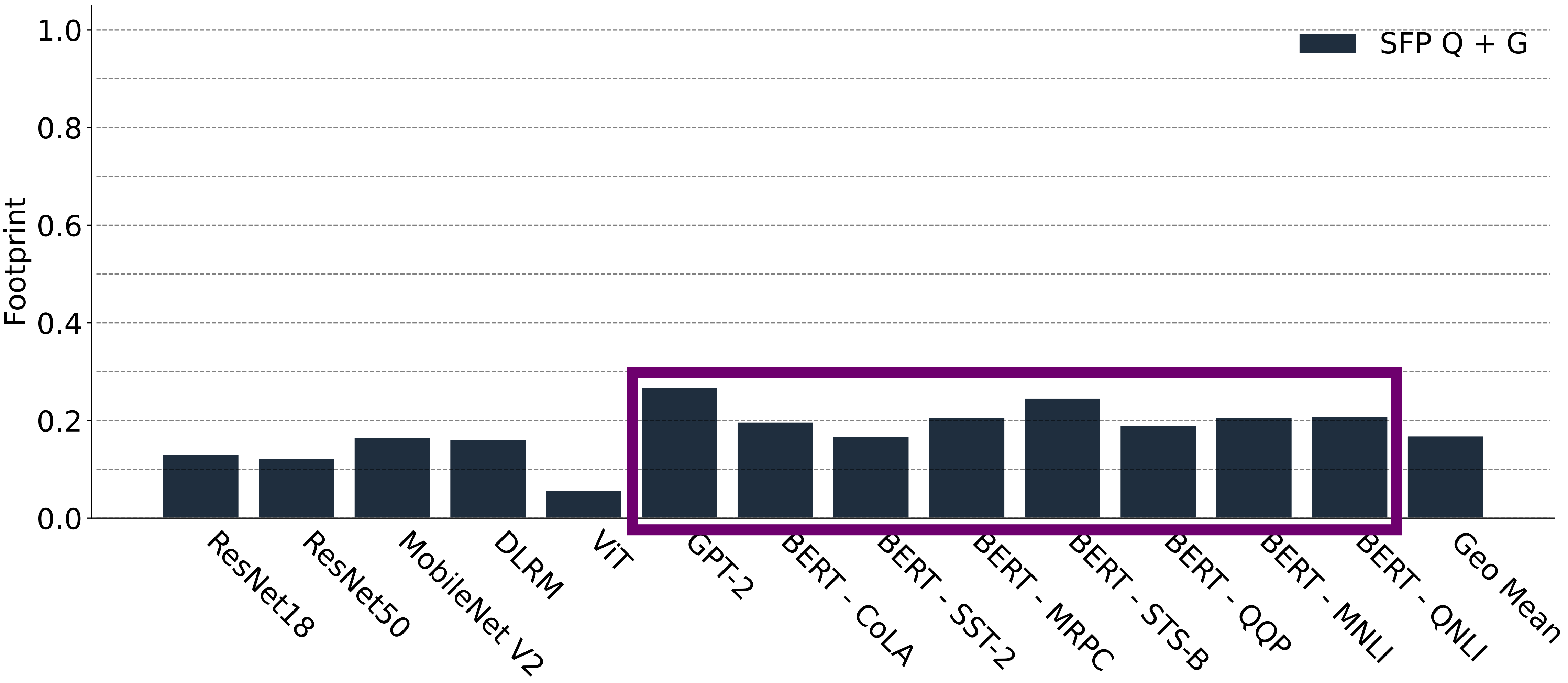
Massive footprint reduction across the board!



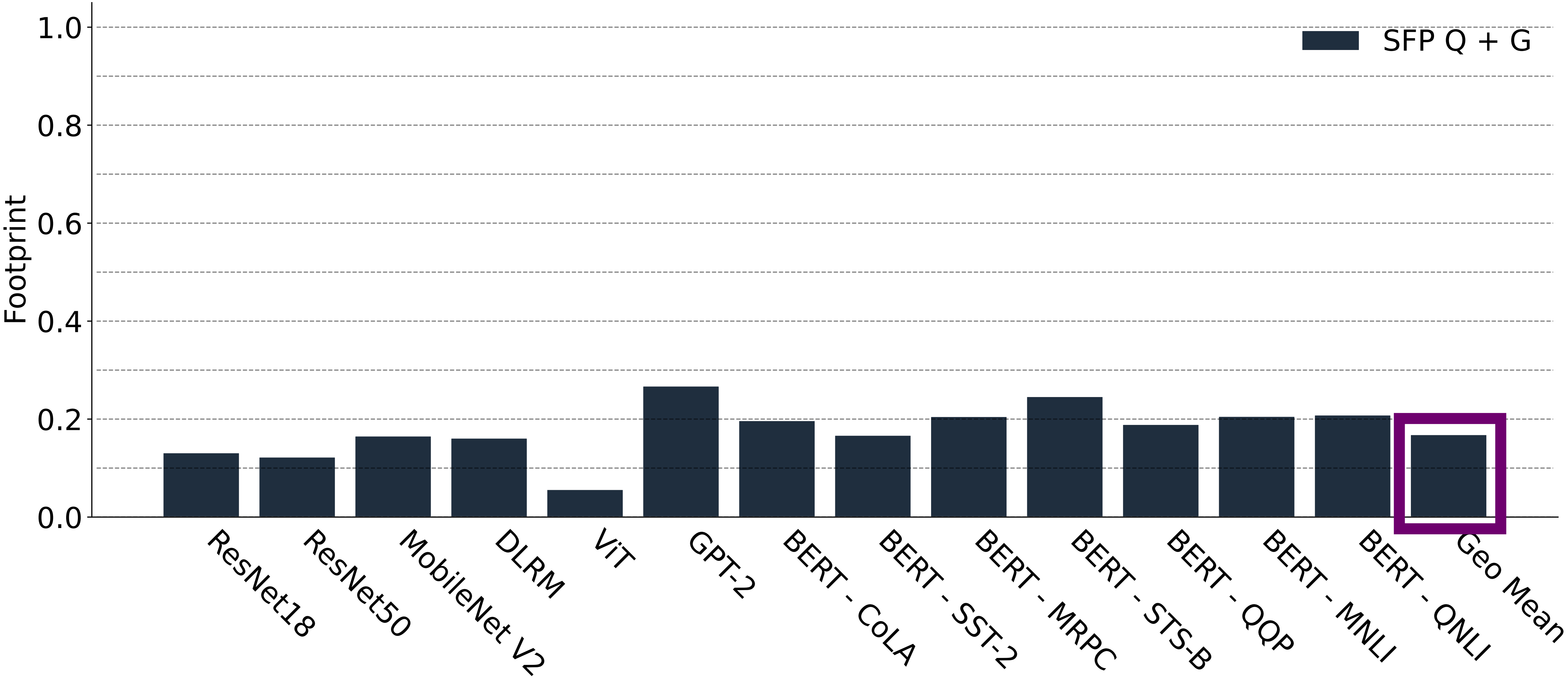
Massive footprint reduction across the board!



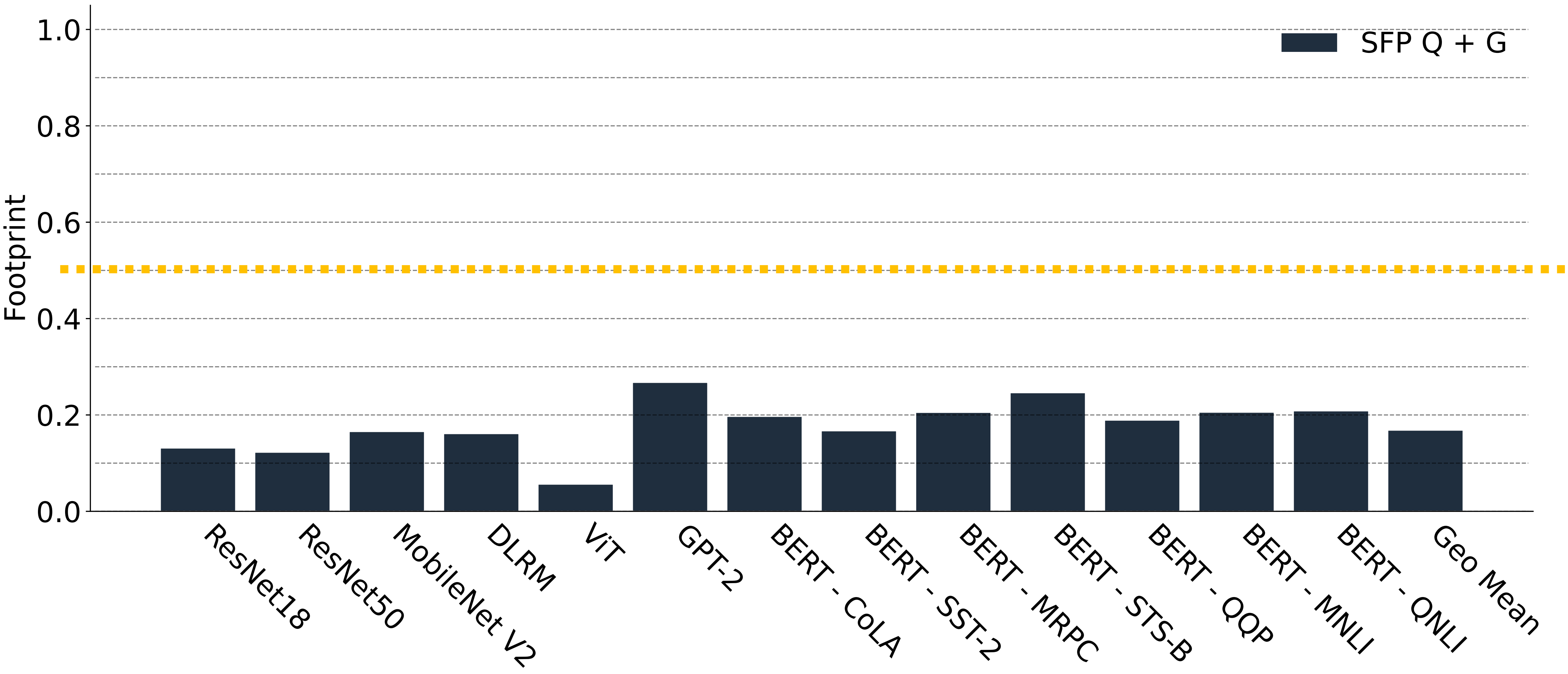
Massive footprint reduction across the board!



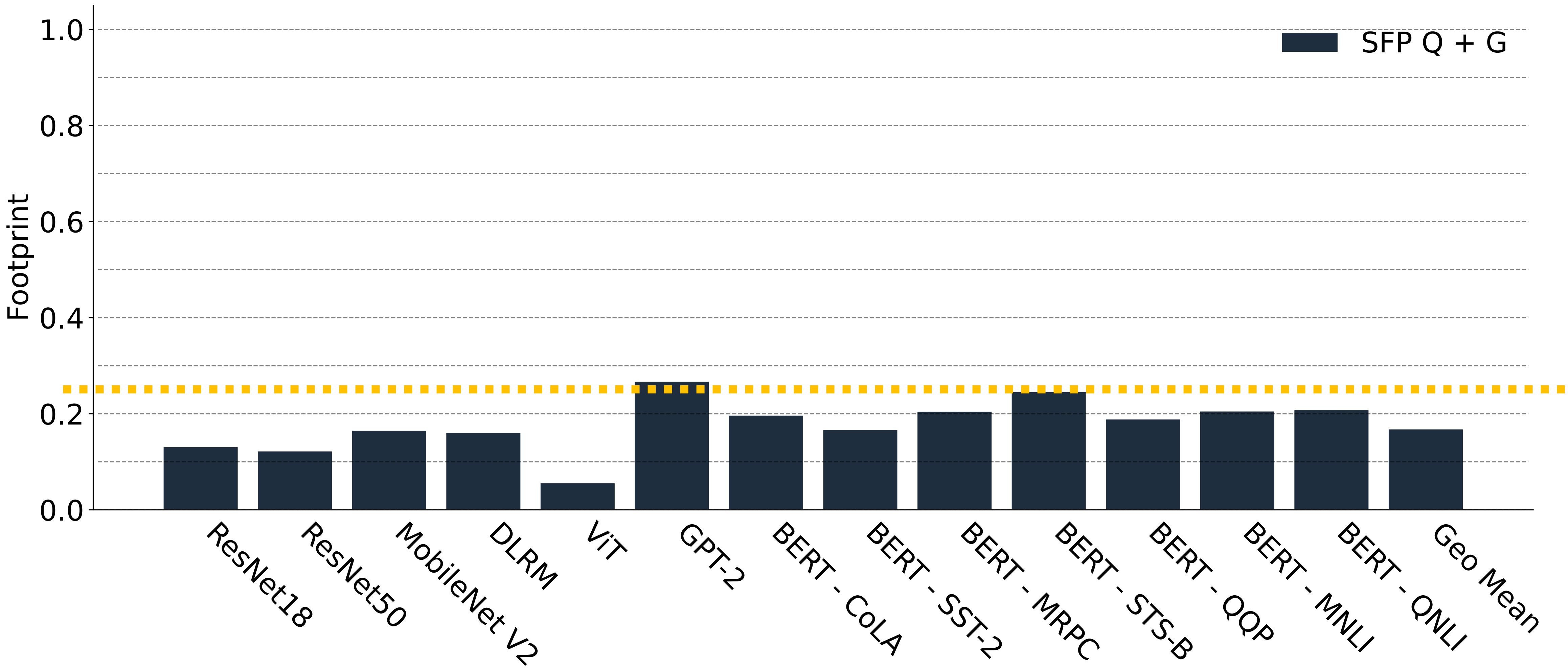
Massive footprint reduction across the board!



Massive footprint reduction across the board!

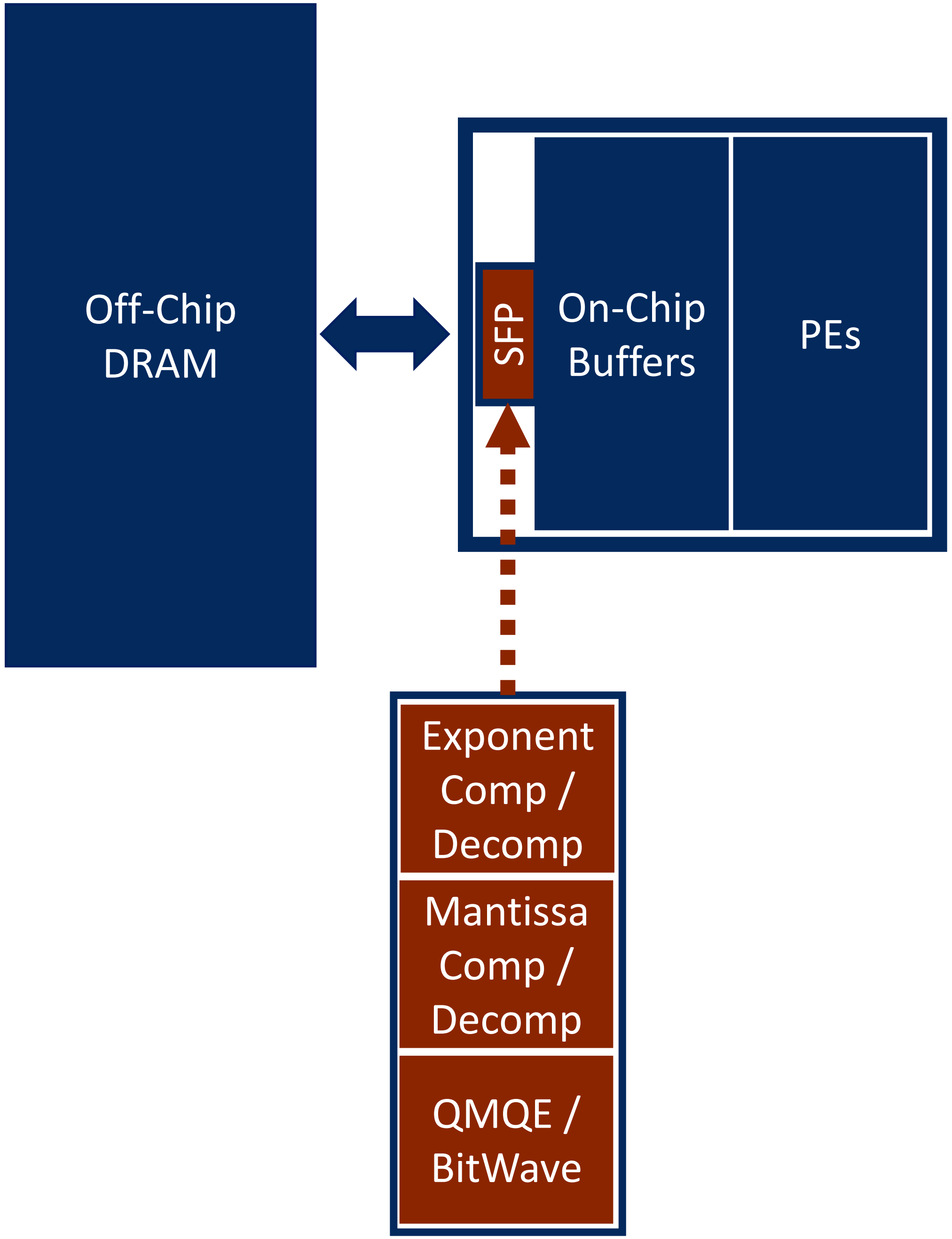


Massive footprint reduction across the board!



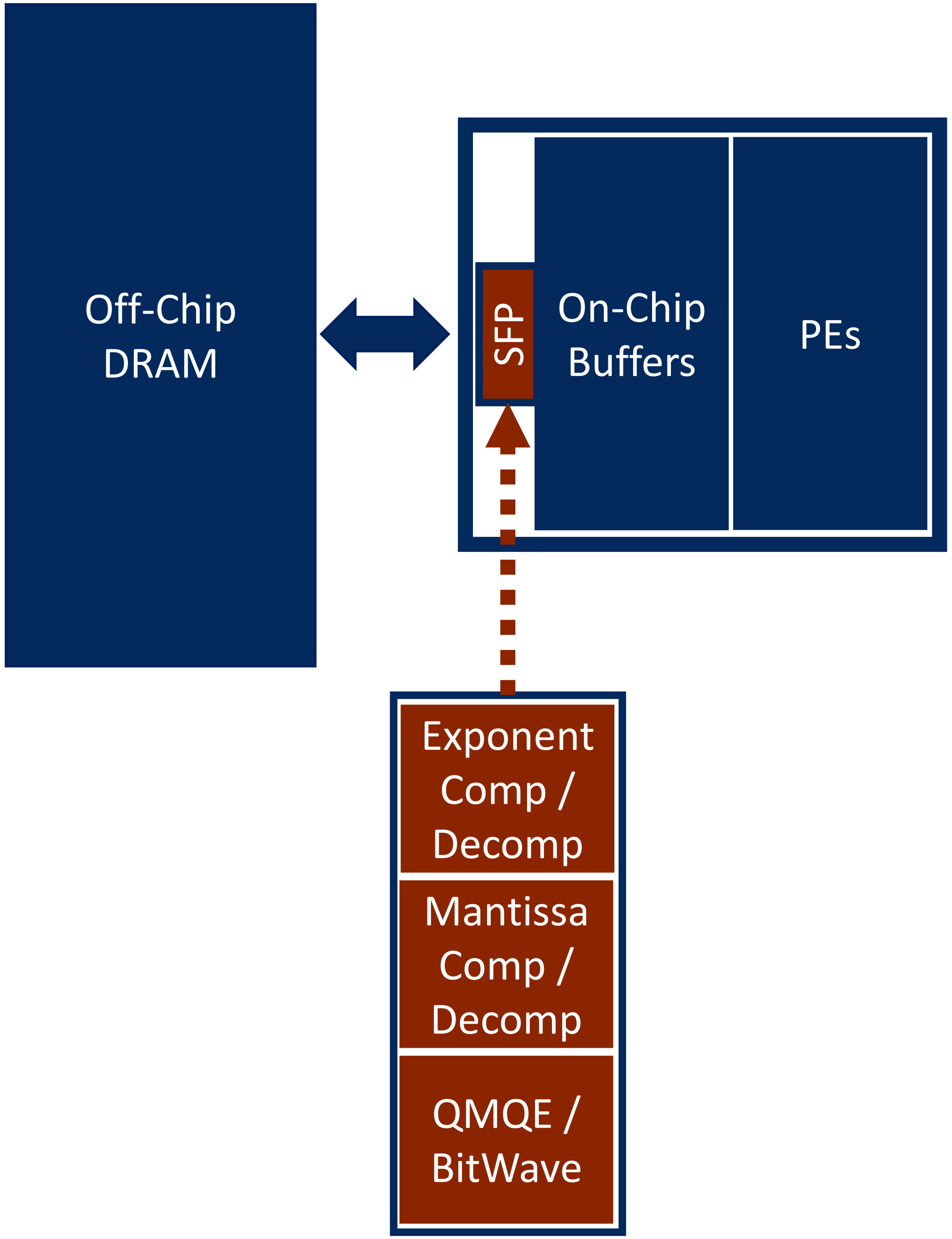
How to exploit?

- Between on- and off-chip memory



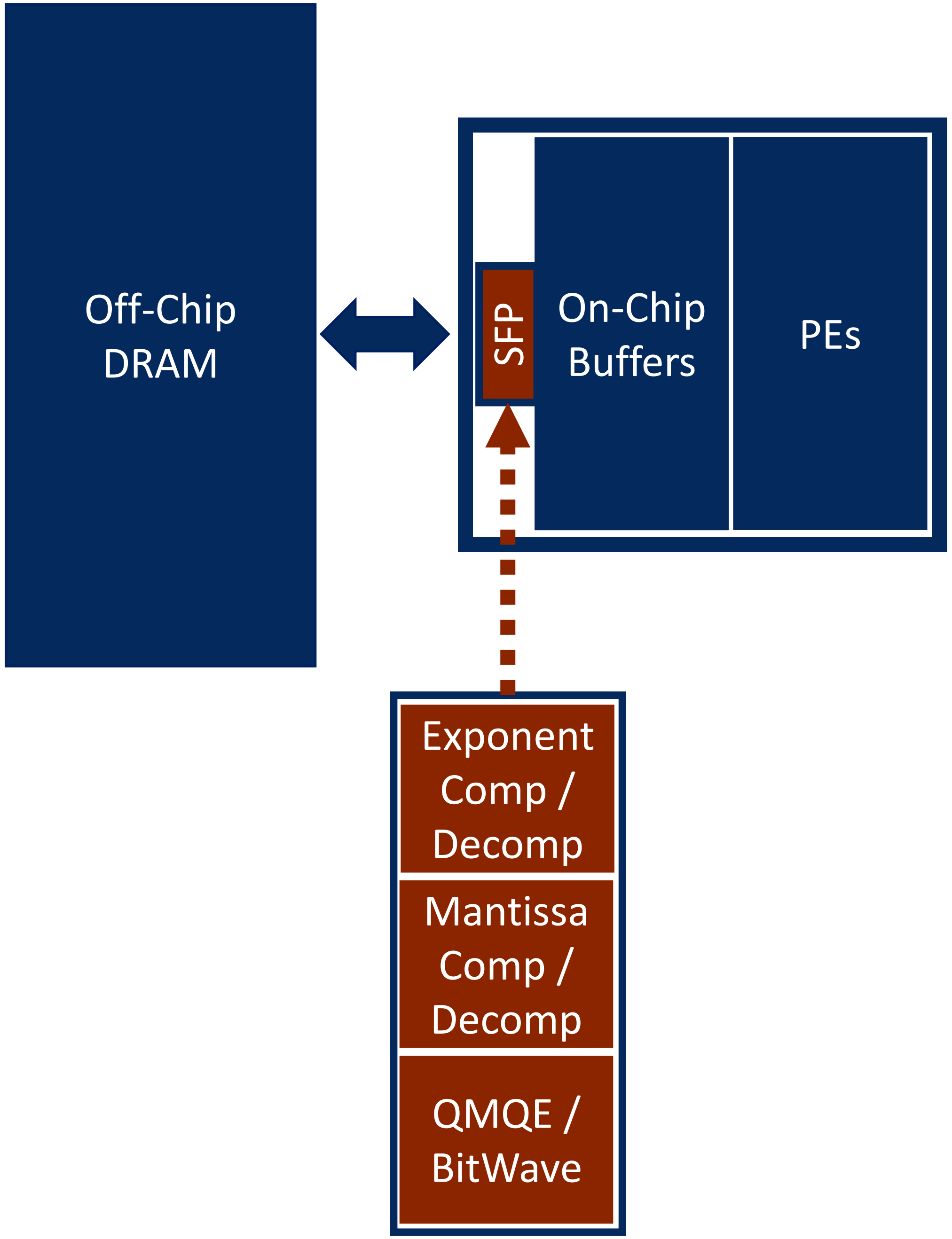
How to exploit?

- Between on- and off-chip memory
- Seamless



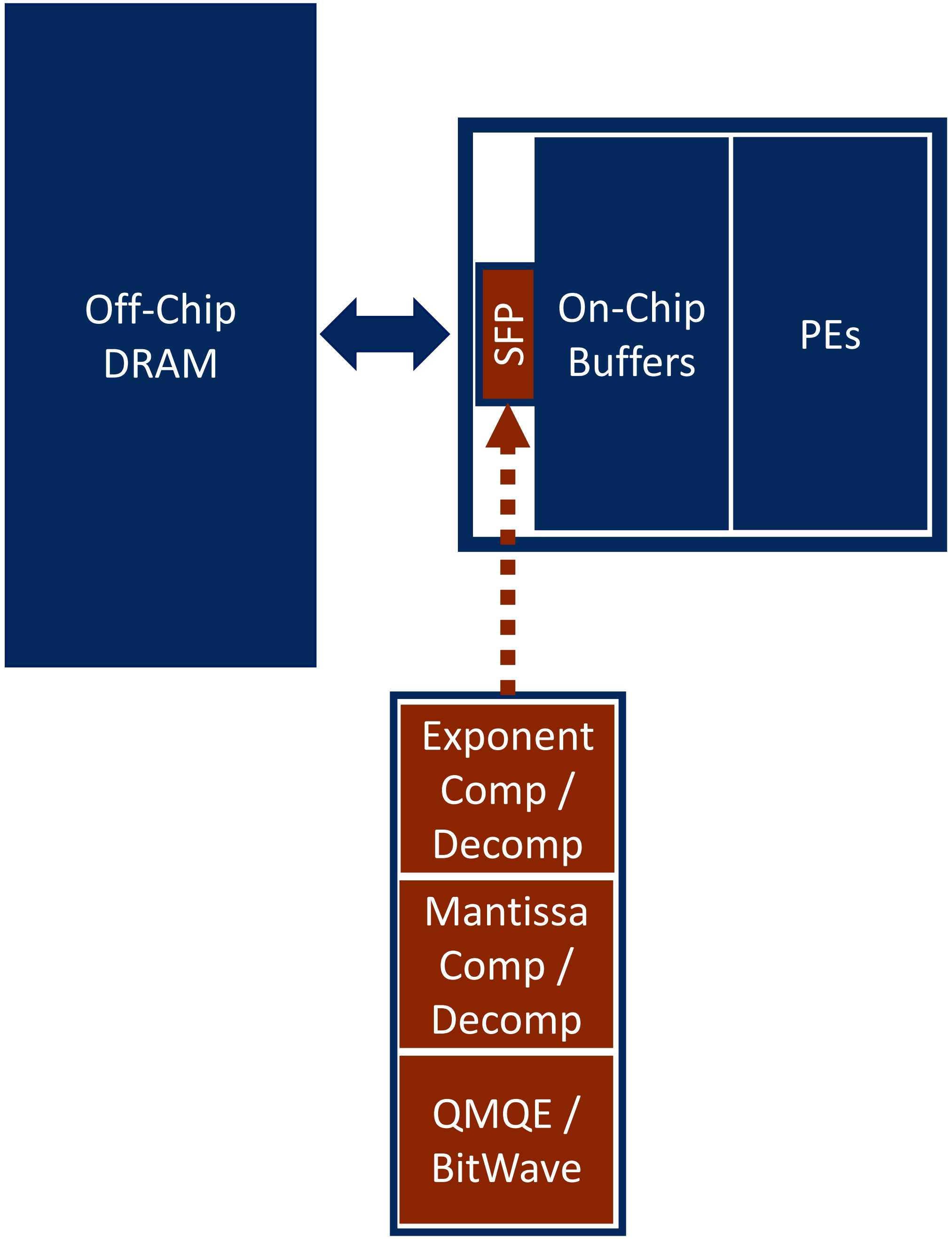
How to exploit?

- Between on- and off-chip memory
- Seamless
- Evaluate in hardware
  - Can be done in software



How to exploit?

- Between on- and off-chip memory
- Seamless
- Evaluate in hardware
  - Can be done in software
- vs. FP8 baseline
  - **2.6x** performance
  - **2.3x** energy efficiency



More in the paper!

- *BitWave* description & results
- Behavior through time
- Results for every network
- Hardware IP blocks
- Simulation results
- Simulation details

Conclusion

- **Schrödinger's FP**: Automatic datatype selection and compression methods

Conclusion

- **Schrödinger's FP**: Automatic datatype selection and compression methods
  - Machine Learning
    - **4.7x** & **5.6x** *(+G)*

Conclusion

- **Schrödinger's FP**: Automatic datatype selection and compression methods
  - Machine Learning
    - **4.7x** & **5.6x** *(+G)*
  - Sampling
    - **3.2x** & **4.6x** (*+G*)

Conclusion

- **Schrödinger's FP**: Automatic datatype selection and compression methods
  - Machine Learning
    - **4.7x** & **5.6x** *(+G)*
  - Sampling
    - **3.2x** & **4.6x** (*+G*)
  - Optionally *+Gecko*

Conclusion

- **Schrödinger's FP**: Automatic datatype selection and compression methods
  - Machine Learning
    - **4.7x** & **5.6x** (*+G*)
  - Sampling
    - **3.2x** & **4.6x** (*+G*)
  - Optionally *+Gecko*

Questions?

Miloš Nikolić <milos.nikolic@mail.utoronto.ca>