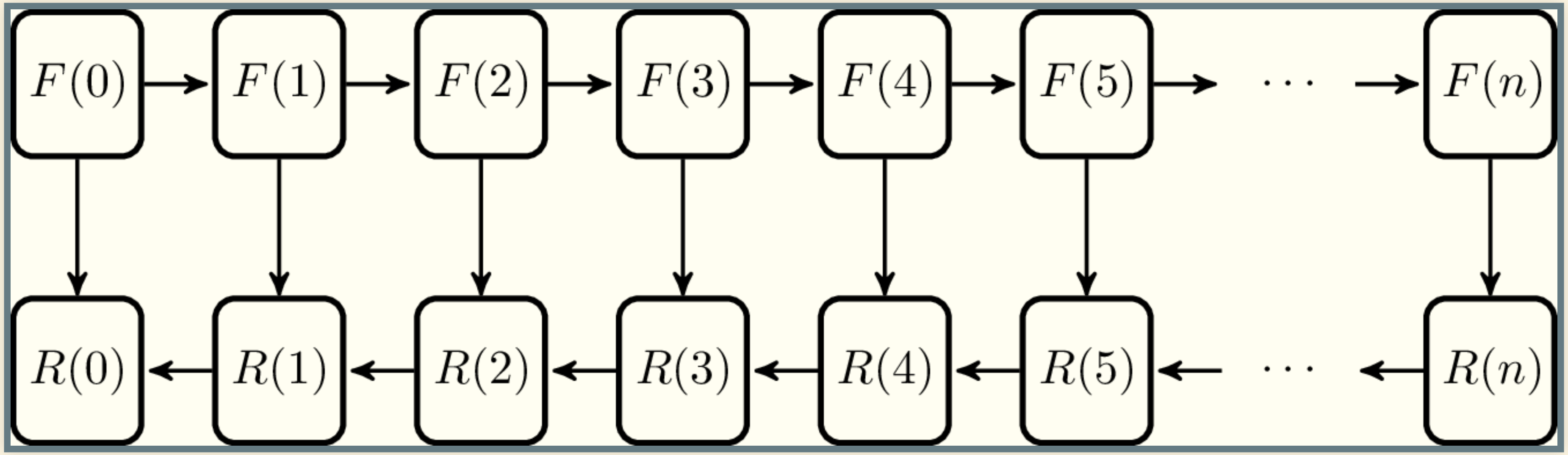
TRAINING NEURAL NETWORKS ON THE EDGE

Navjot Kukreja, Alena Shilova

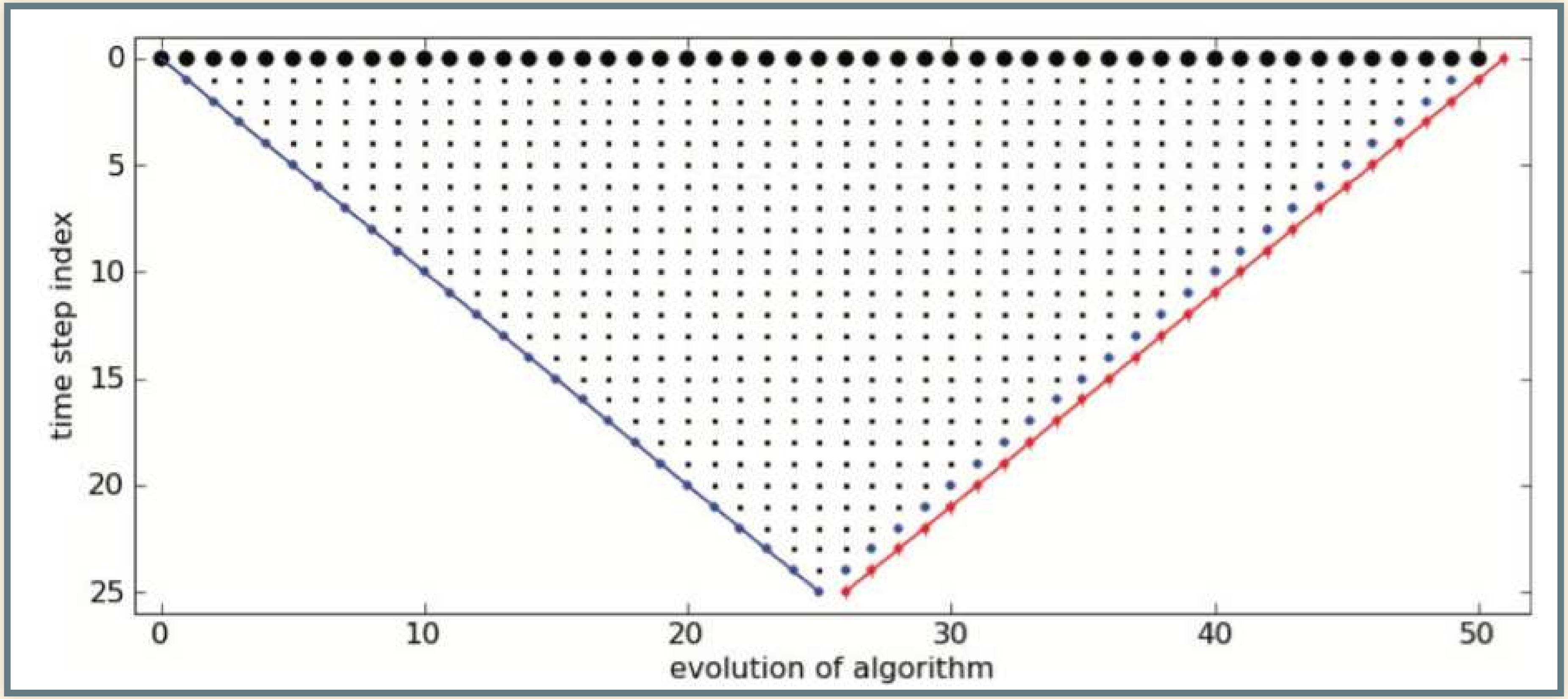
Also:

- Olivier Beaumont
- Jan Huckelheim
- Nicola Ferrier
- Paul Hovland
- Gerard Gorman

BACKGROUND



Typical data flow pattern for adjoint problems



Memory consumption during an adjoint problem

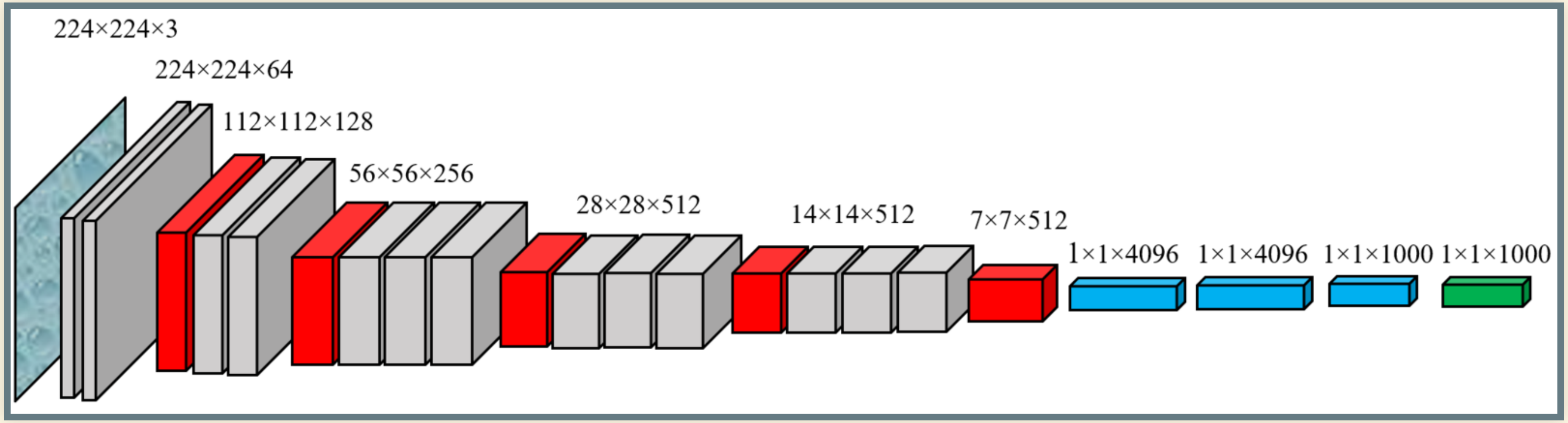
Checkpointing (Revolve)

Setup

▷ Forward step	▶ Executing forward step	▶ Saved forward step	◁ Reverse step	◀ Executing reverse step	◀ Reverse step completed

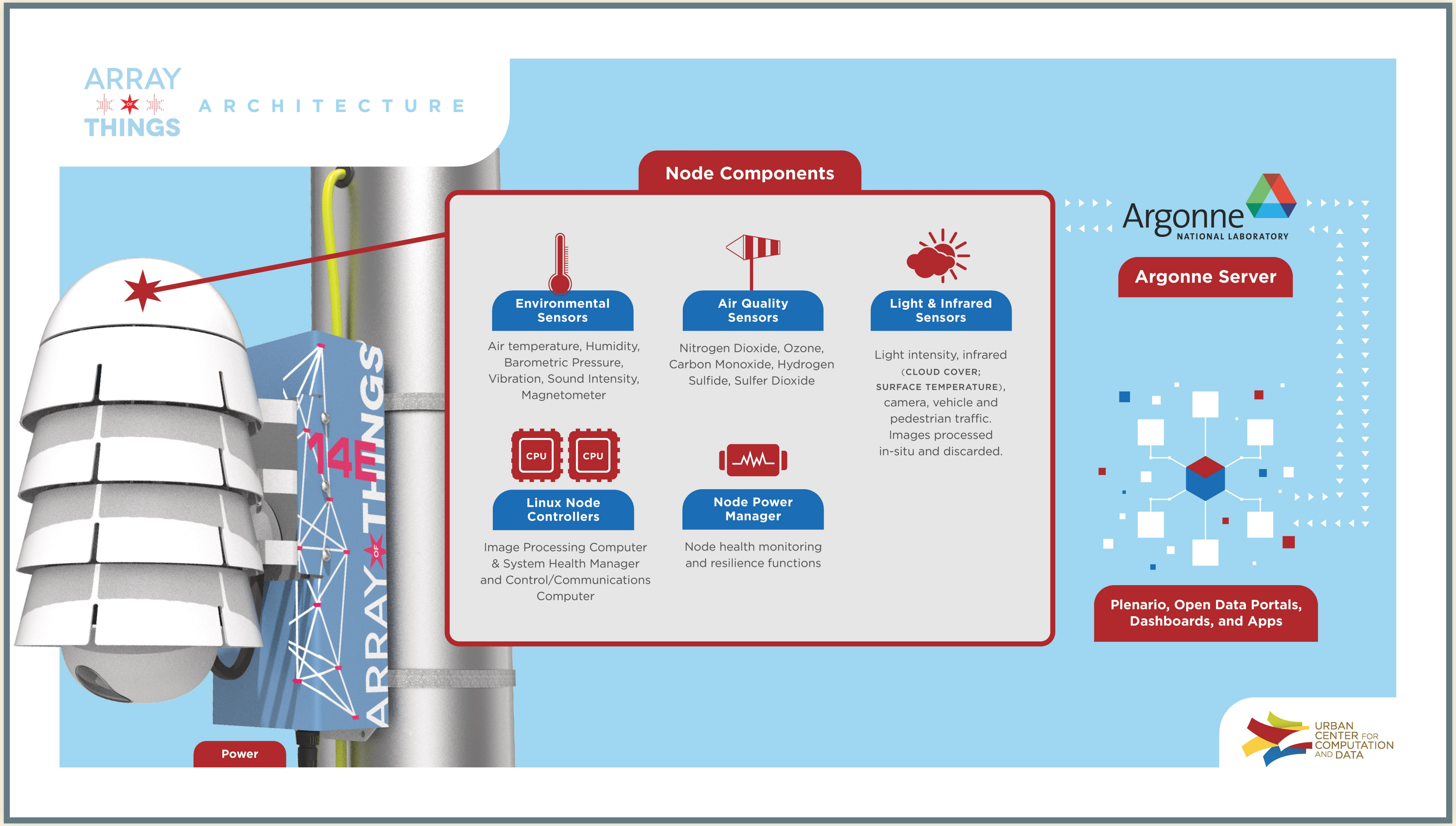
.

Where else do we see the same data-access pattern?



VGGNet

ARRAY OF THINGS



ARRAY OF THINGS ARCHITECTURE

Node Components

Environmental Sensors

Air temperature, Humidity, Barometric Pressure, Vibration, Sound Intensity, Magnetometer

Air Quality Sensors

Nitrogen Dioxide, Ozone, Carbon Monoxide, Hydrogen Sulfide, Sulfer Dioxide

Light & Infrared Sensors

Light intensity, infrared (CLOUD COVER; SURFACE TEMPERATURE), camera, vehicle and pedestrian traffic. Images processed in-situ and discarded.

Linux Node Controllers

Image Processing Computer & System Health Manager and Control/Communications Computer

Node Power Manager

Node health monitoring and resilience functions

Argonne Server

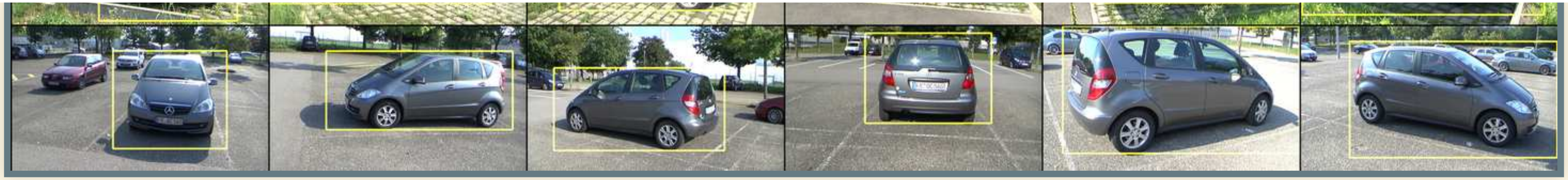
Plenario, Open Data Portals, Dashboards, and Apps

CPU
CPU
14E
ARRAY OF THINGS
Power
Argonne NATIONAL LABORATORY
URBAN CENTER FOR COMPUTATION AND DATA

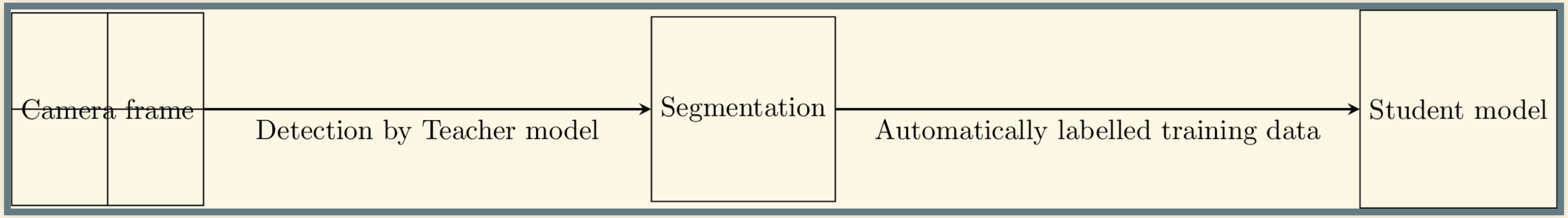
WAGGLE PAYLOAD COMPUTER

- ODROID XU4 based on the Samsung Exynos5422 CPU
- four A15 cores, four A7 cores
- Mali-T628 MP6 GPU that supports OpenCL, 2GB LPDDR3 RAM
- attached flash storage

VIEWPOINT PROBLEM



STUDENT-TEACHER MODEL

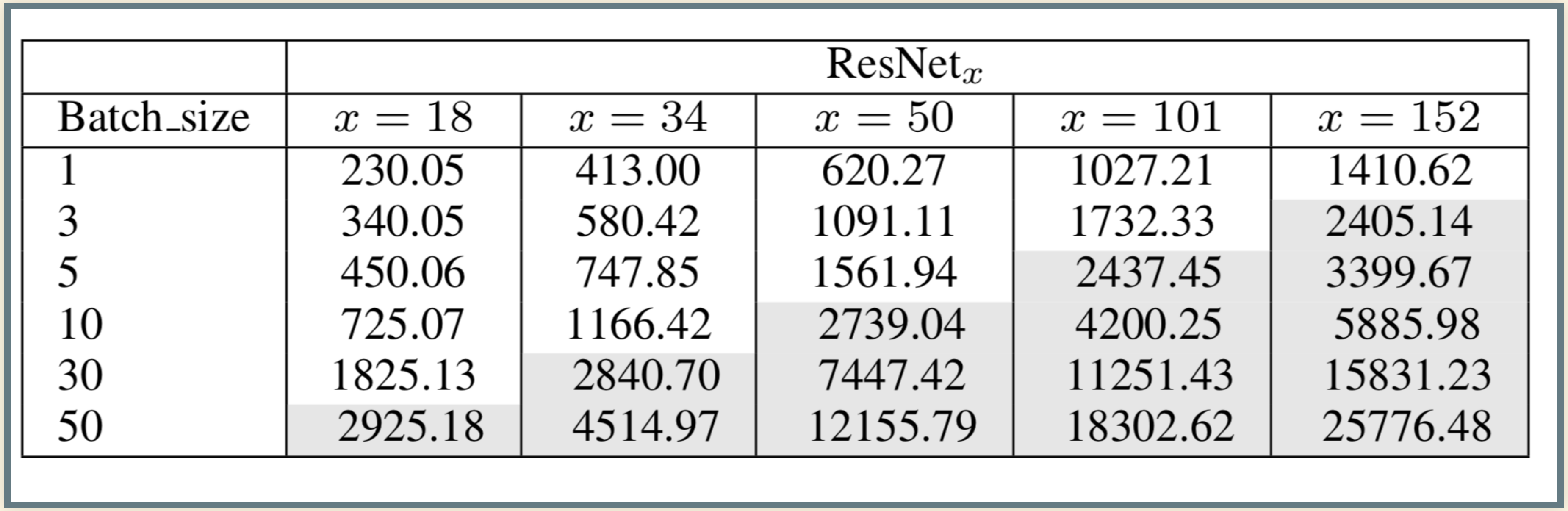


Camera frame
Detection by Teacher model
Segmentation
Automatically labelled training data
Student model

CHALLENGES

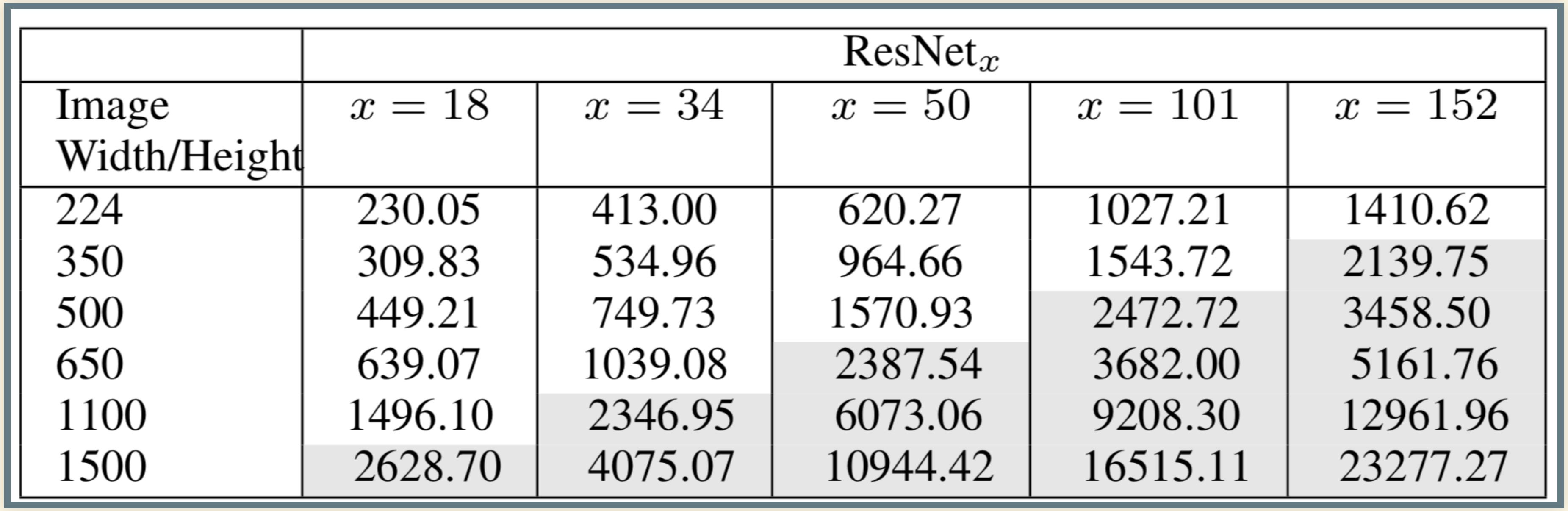
- Network (not a challenge)
- Storage (not a challenge)
- Computation (not necessarily a challenge)
- Memory!

MEMORY REQUIRED TO TRAIN RESNET



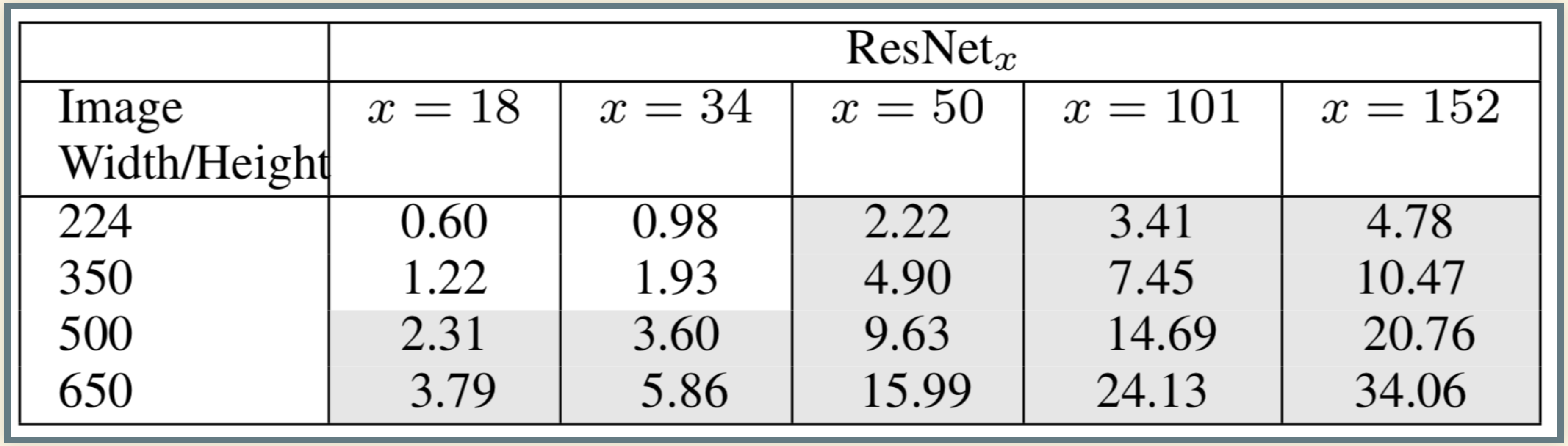
	ResNet$_x$				
Batch_size	$x = 18$	$x = 34$	$x = 50$	$x = 101$	$x = 152$
1	230.05	413.00	620.27	1027.21	1410.62
3	340.05	580.42	1091.11	1732.33	2405.14
5	450.06	747.85	1561.94	2437.45	3399.67
10	725.07	1166.42	2739.04	4200.25	5885.98
30	1825.13	2840.70	7447.42	11251.43	15831.23
50	2925.18	4514.97	12155.79	18302.62	25776.48

Memory required (MB) for image size 224×224



	ResNet$_x$				
Image Width/Height	$x = 18$	$x = 34$	$x = 50$	$x = 101$	$x = 152$
224	230.05	413.00	620.27	1027.21	1410.62
350	309.83	534.96	964.66	1543.72	2139.75
500	449.21	749.73	1570.93	2472.72	3458.50
650	639.07	1039.08	2387.54	3682.00	5161.76
1100	1496.10	2346.95	6073.06	9208.30	12961.96
1500	2628.70	4075.07	10944.42	16515.11	23277.27

Memory required (MB) for batch size 1



	ResNet$_x$				
Image Width/Height	$x = 18$	$x = 34$	$x = 50$	$x = 101$	$x = 152$
224	0.60	0.98	2.22	3.41	4.78
350	1.22	1.93	4.90	7.45	10.47
500	2.31	3.60	9.63	14.69	20.76
650	3.79	5.86	15.99	24.13	34.06

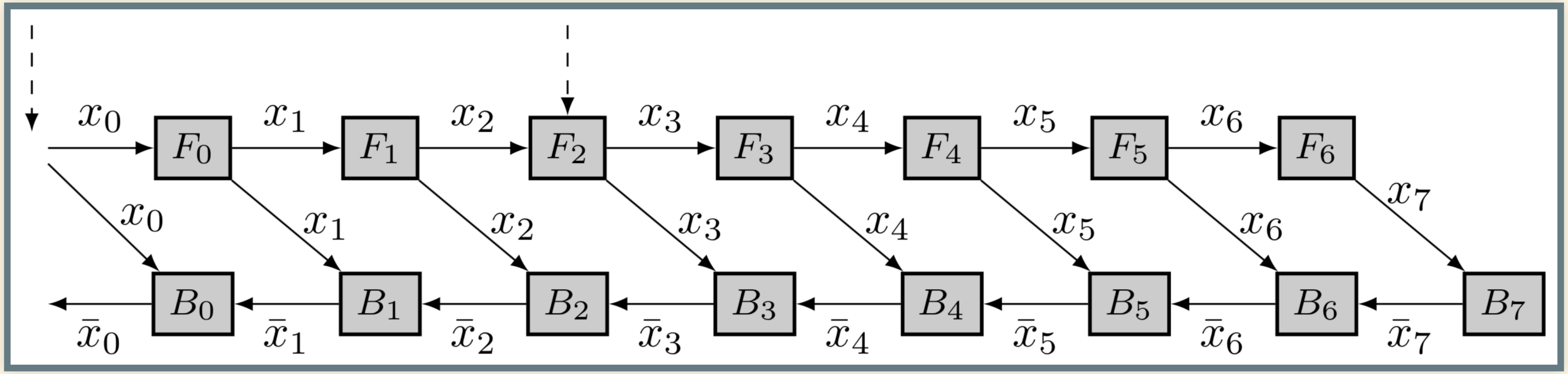
Memory required (GB) for batch size 8

CHECKPOINTING

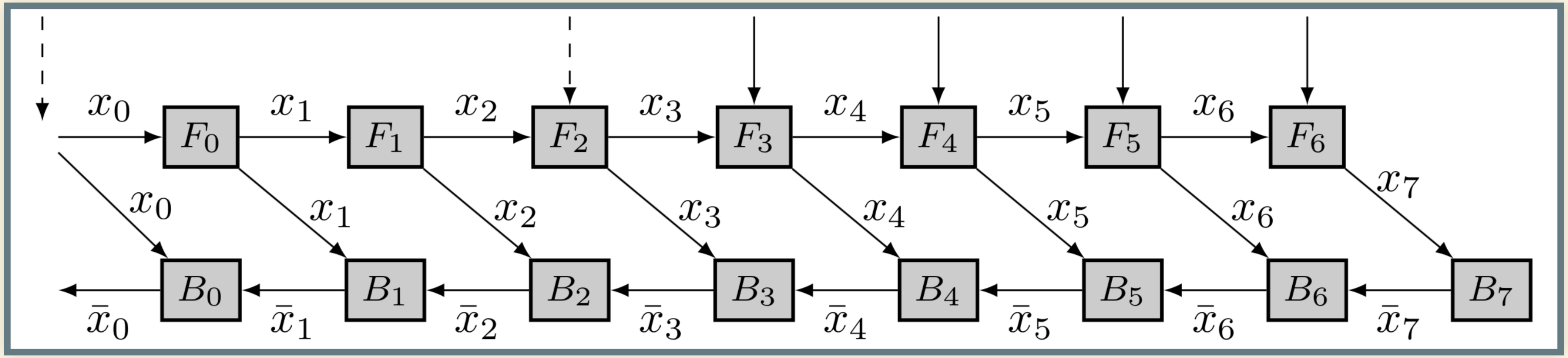
PyTorch

- fast-evolving Python package widely applied in deep learning
- uses Tensors as a basic class
- Tensors are similar to NumPy arrays which also allow to work with them on GPU
- dynamically defines the computational graph of the model
- designed to be memory efficient: there is checkpointing strategy

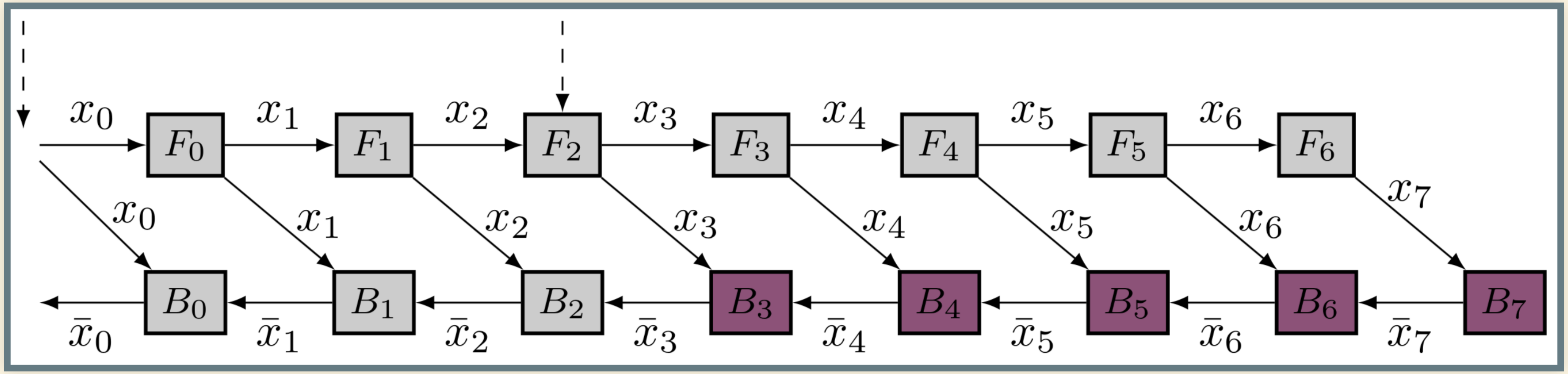
Checkpoint sequential: number of segments = 2



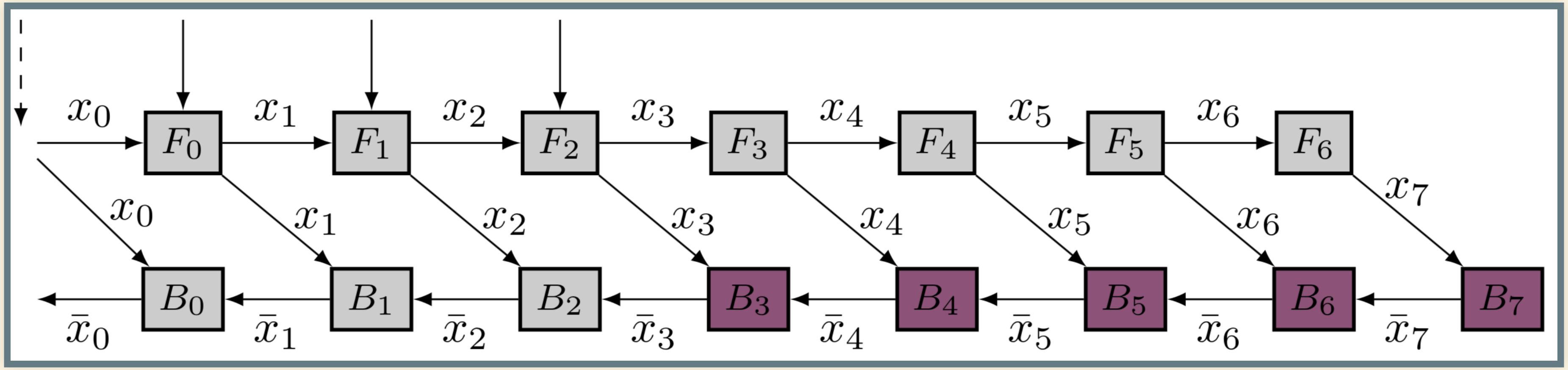
Checkpoint sequential: number of segments = 2



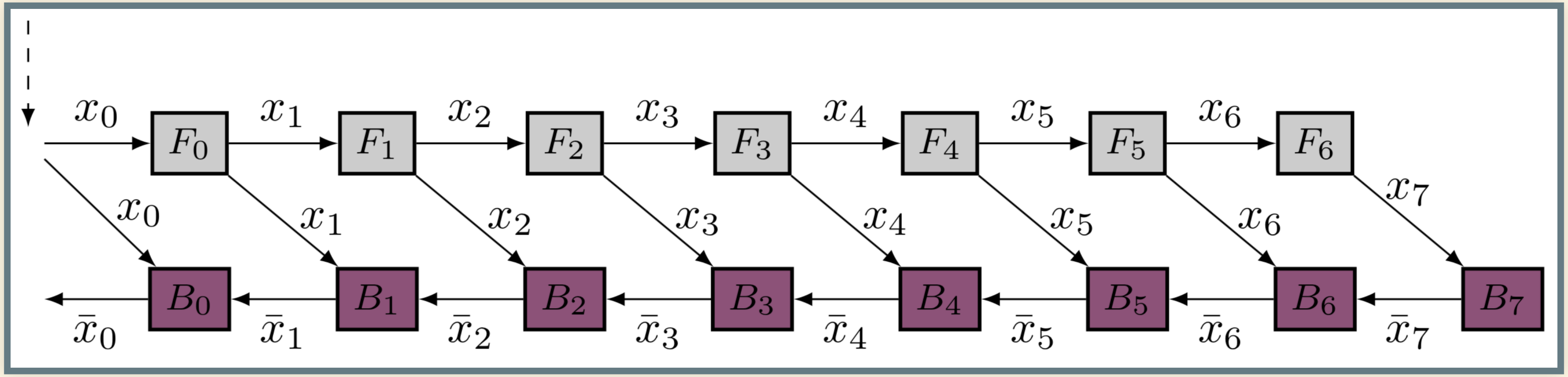
Checkpoint sequential: number of segments = 2



Checkpoint sequential: number of segments = 2

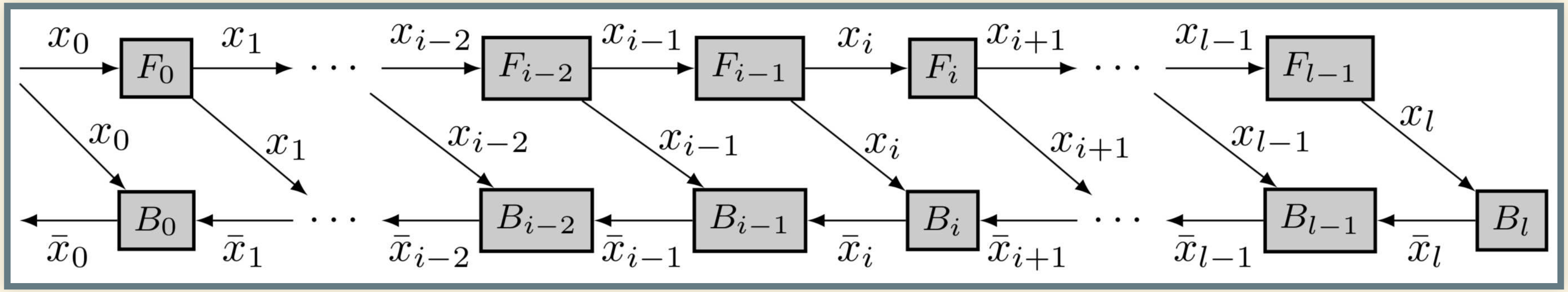


Checkpoint sequential: number of segments = 2



$$ \mbox{Memory} = s - 1 + \bigl(l - \left\lfloor l/s \right\rfloor (s -1) \bigr). $$

Revolve: dynamic programming

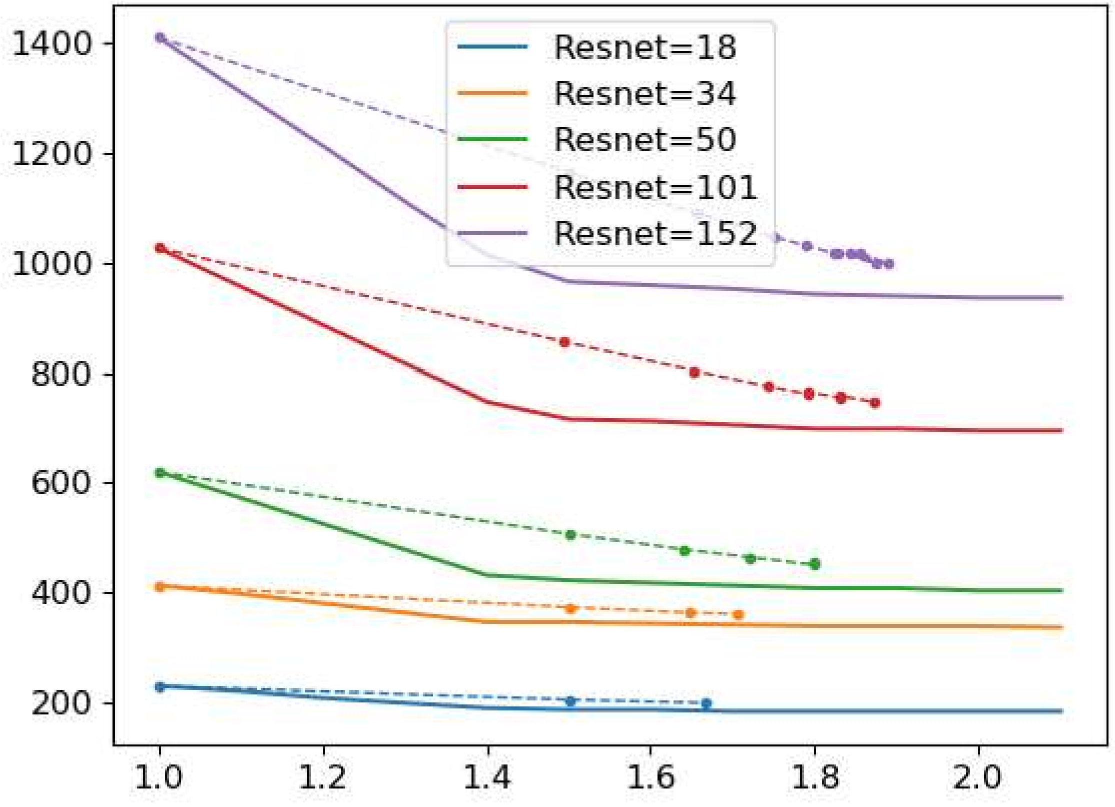


$$ \small{\mbox{Opt}[\ell,1] = \frac{\ell (\ell +1)}{2} u_f + (\ell+1) u_b}$$

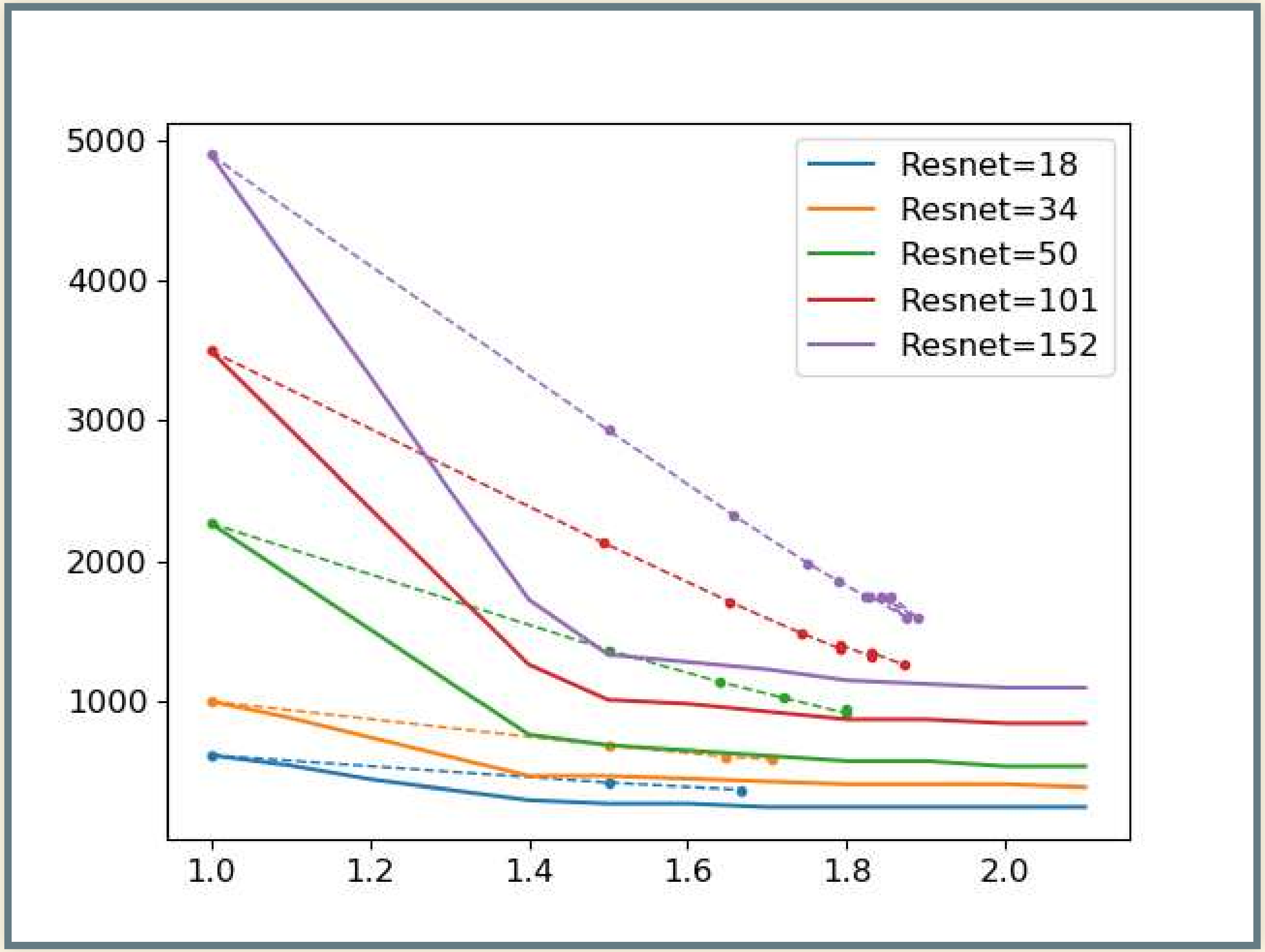
$$\small{\mbox{Opt}[1, c] = u_f +2 u_b}$$

$$\small{\mbox{Opt}[\ell, c] = \min_{1 \leq i \leq \ell-1} (i u_f +\mbox{Opt}[\ell - i, c -1] + \mbox{Opt}[i-1, c]) }$$

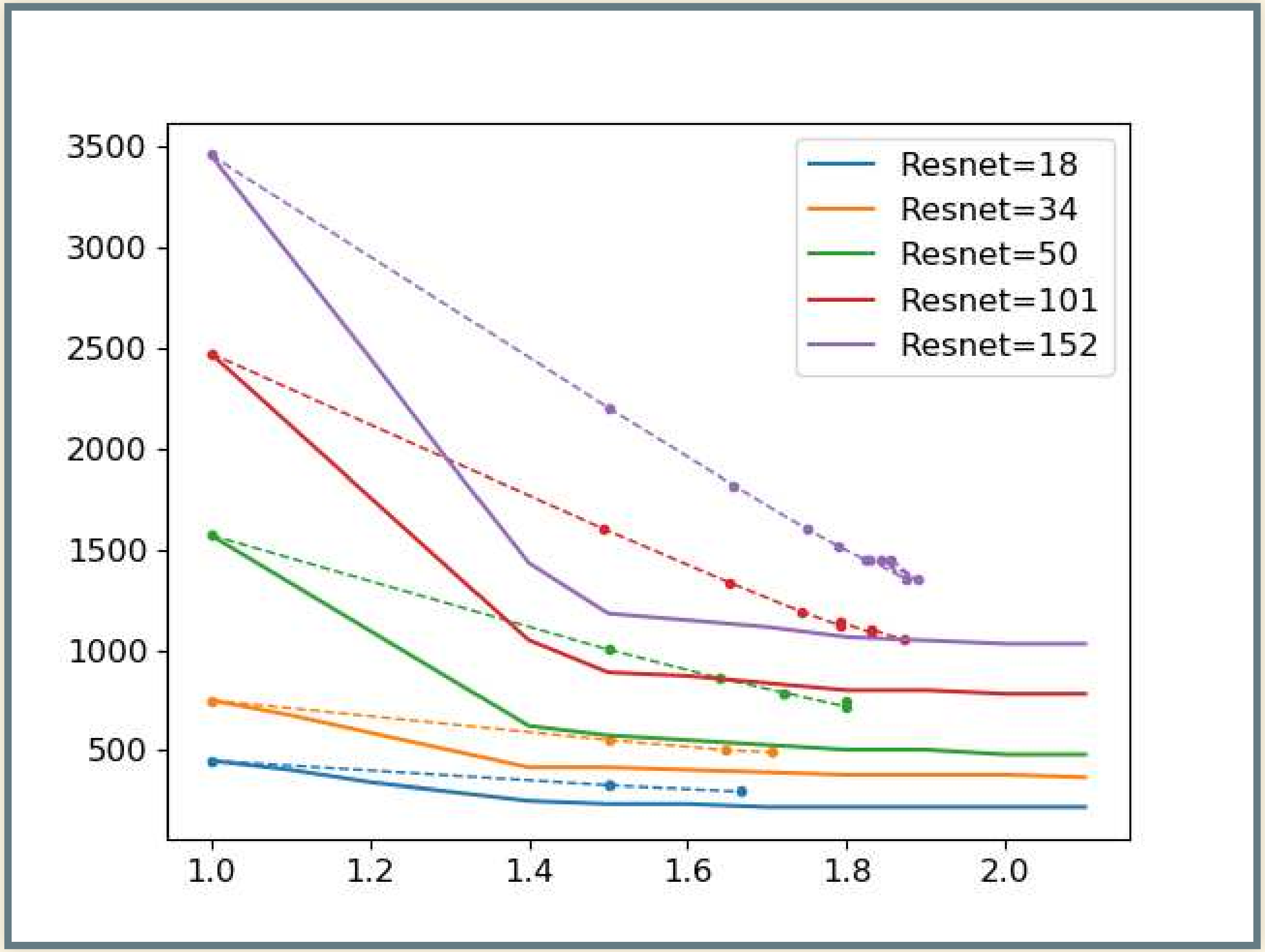
Comparison of Checkpoint sequential and Revolve



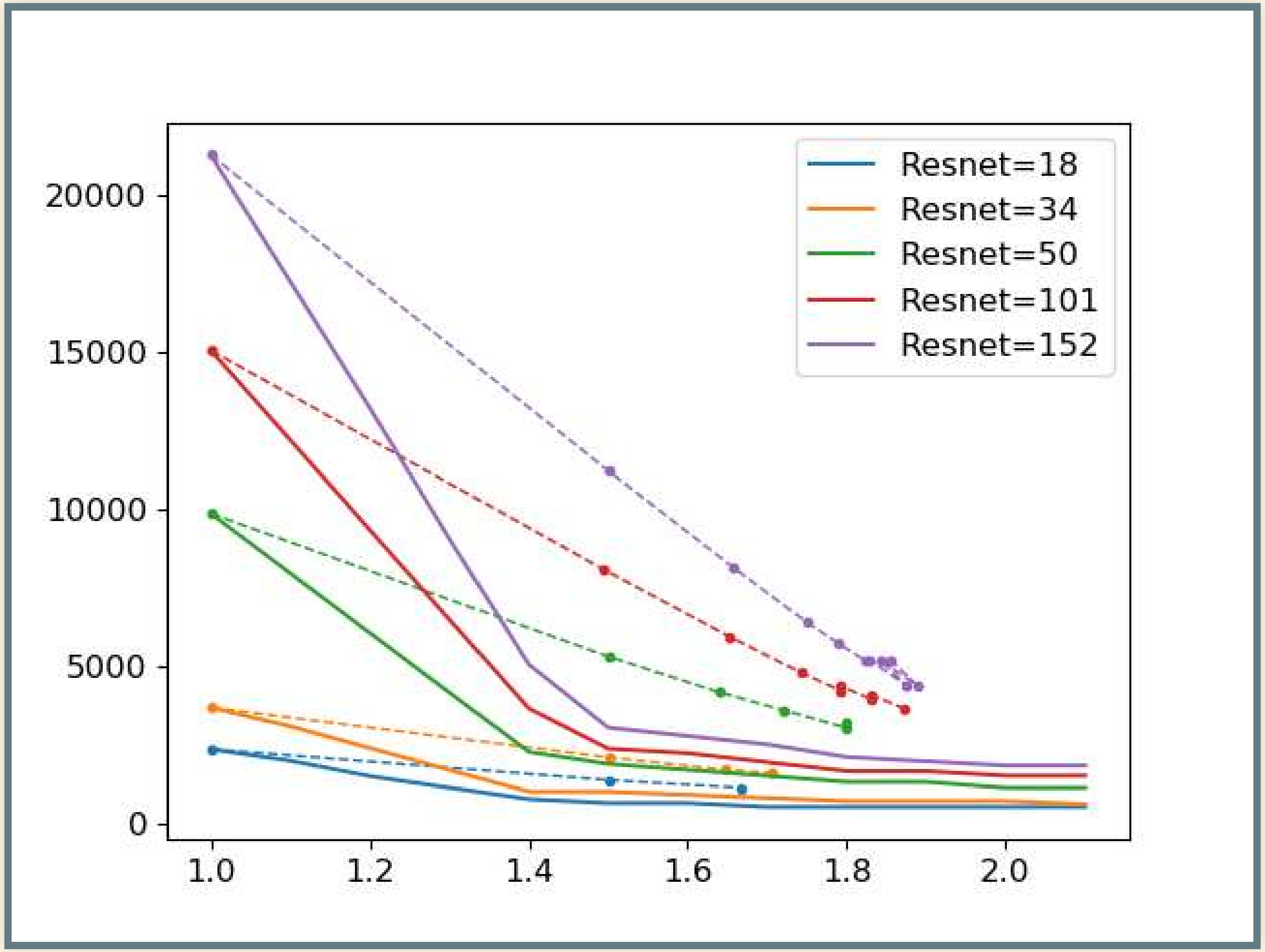
Batch Size: 1, Image Size: 224×224



Batch Size: 8, Image Size: 224×224



Batch Size: \$1\$, Image Size: \$500 \times 500\$



Batch Size: 8, Image Size: 500×500

PRACTICAL IMPLEMENTATION AND CONCLUDING REMARKS

THANK YOU