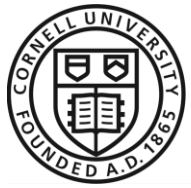
EVA^2: Exploiting Temporal Redundancy In Live Computer Vision

Mark Buckler, Philip Bedoukian, Suren Jayasuriya, Adrian Sampson



Cornell University

ASU Arizona State University

International Symposium on Computer Architecture (ISCA)

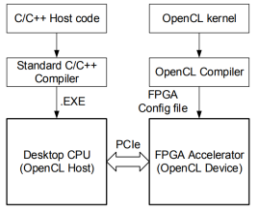
Tuesday June 5, 2018

Convolutional Neural Networks (CNNs)

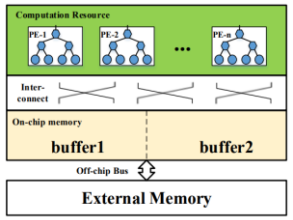
Convolutional Neural Networks (CNNs)

Embedded Vision Accelerators

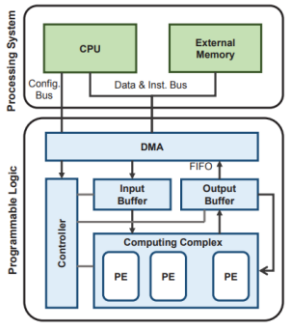
FPGA Research



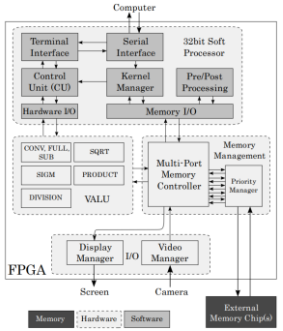
Suda et al.



Zhang et al.



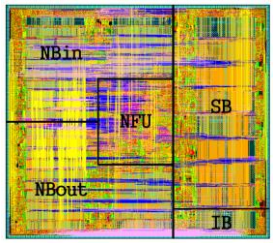
Qiu et al.



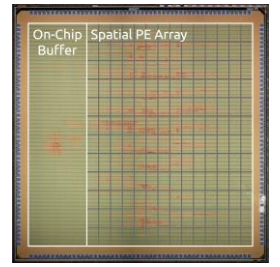
Farabet et al.

Many more…

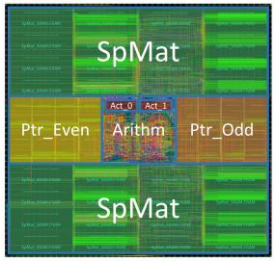
ASIC Research



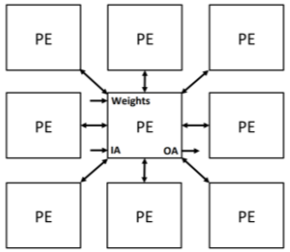
ShiDianNao



Eyeriss



EIE



SCNN

Many more…

Industry Adoption

Microsoft

NVIDIA

Google

CEVA

HUAWEI

Movidius

Temporal Redundancy

Frame 0

Frame 1

Frame 2

Frame 3

Input Change	**High**	**Low**	**Low**	**Low**

Temporal Redundancy

Frame 0

Frame 1

Frame 2

Frame 3

Input Change	**High**	**Low**	**Low**	**Low**
Cost to Process	**High**	**High**	**High**	**High**

Temporal Redundancy

Frame 0

Frame 1

Frame 2

Frame 3

Input Change	**High**	**Low**	**Low**	**Low**
Cost to Process	**High**	~~**High**~~ **Low**	~~**High**~~ **Low**	~~**High**~~ **Low**

Talk Overview

Background

Algorithm

Hardware

Evaluation

Conclusion

Talk Overview

Background

Algorithm

Hardware

Evaluation

Conclusion

Common Structure in CNNs

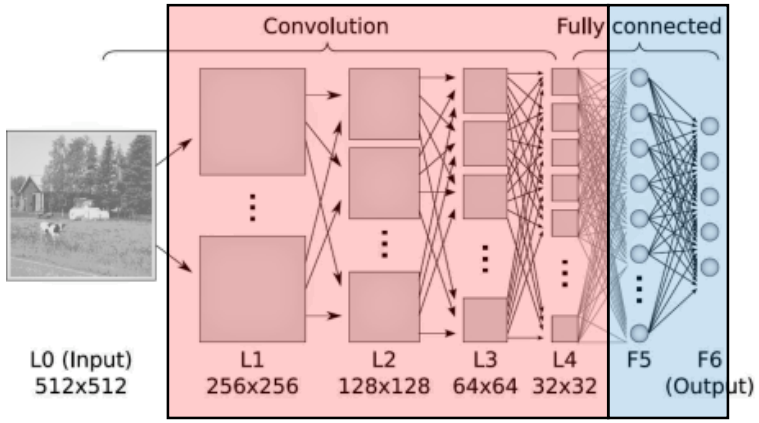
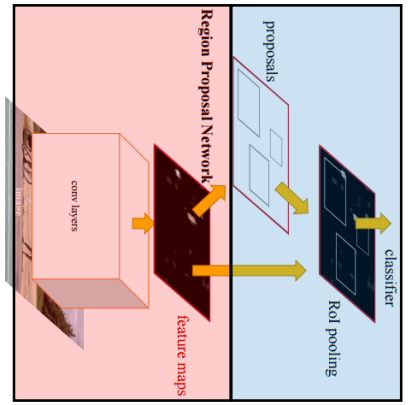
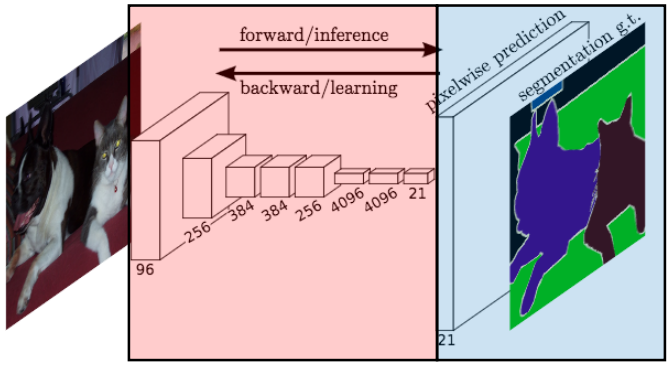


Image Classification



Object Detection



Semantic Segmentation

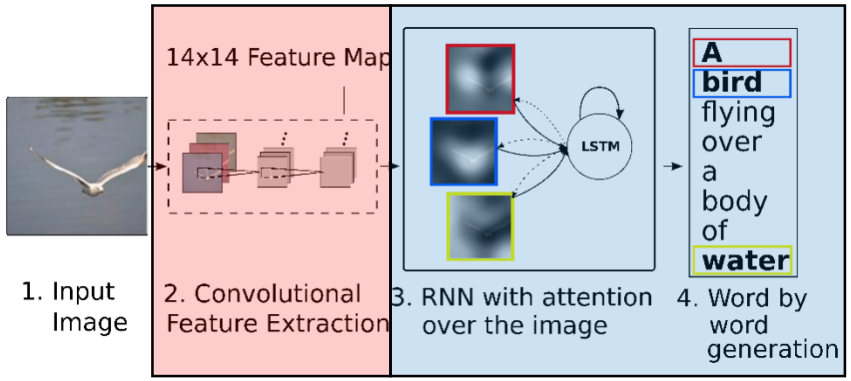
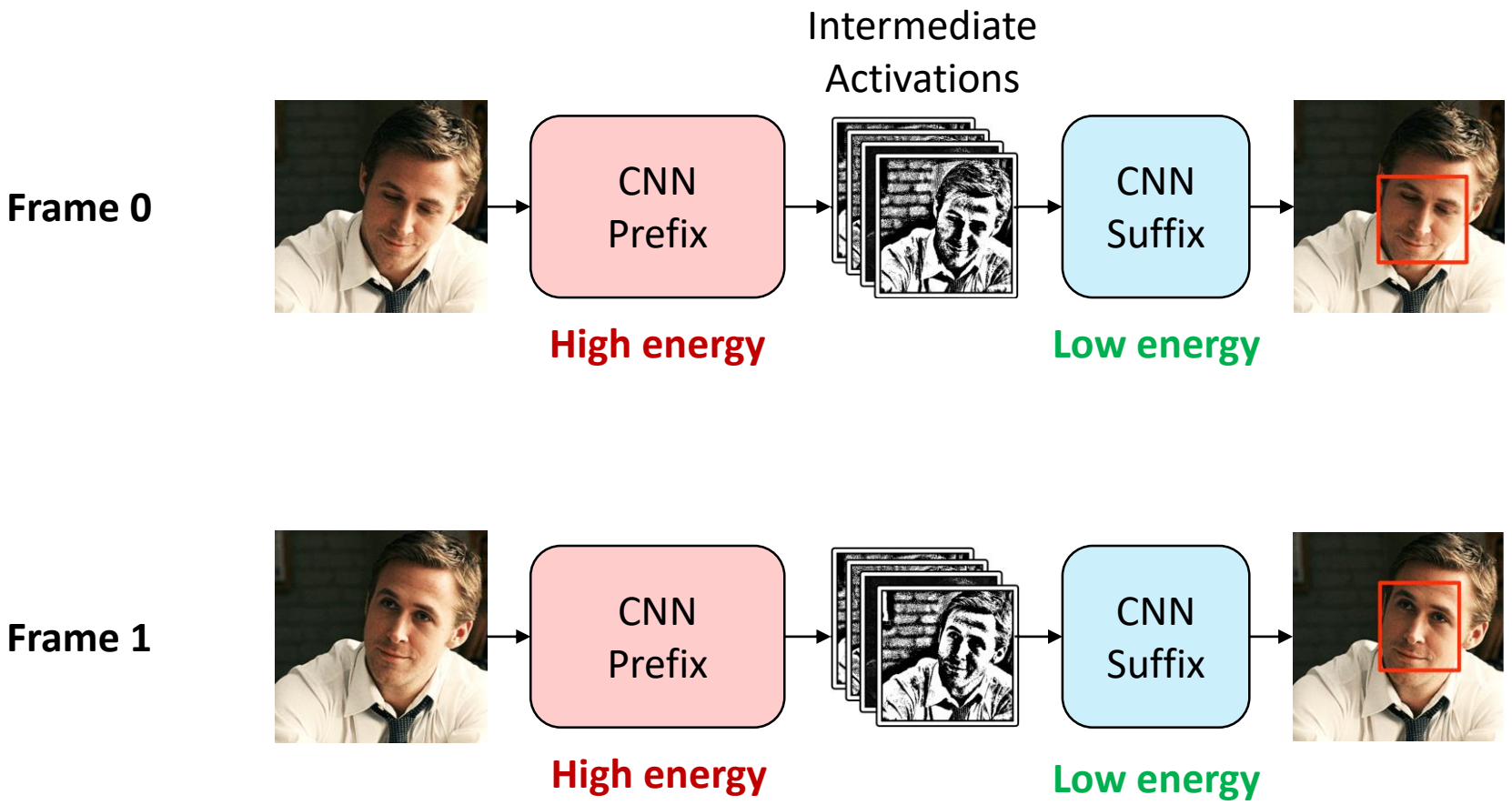


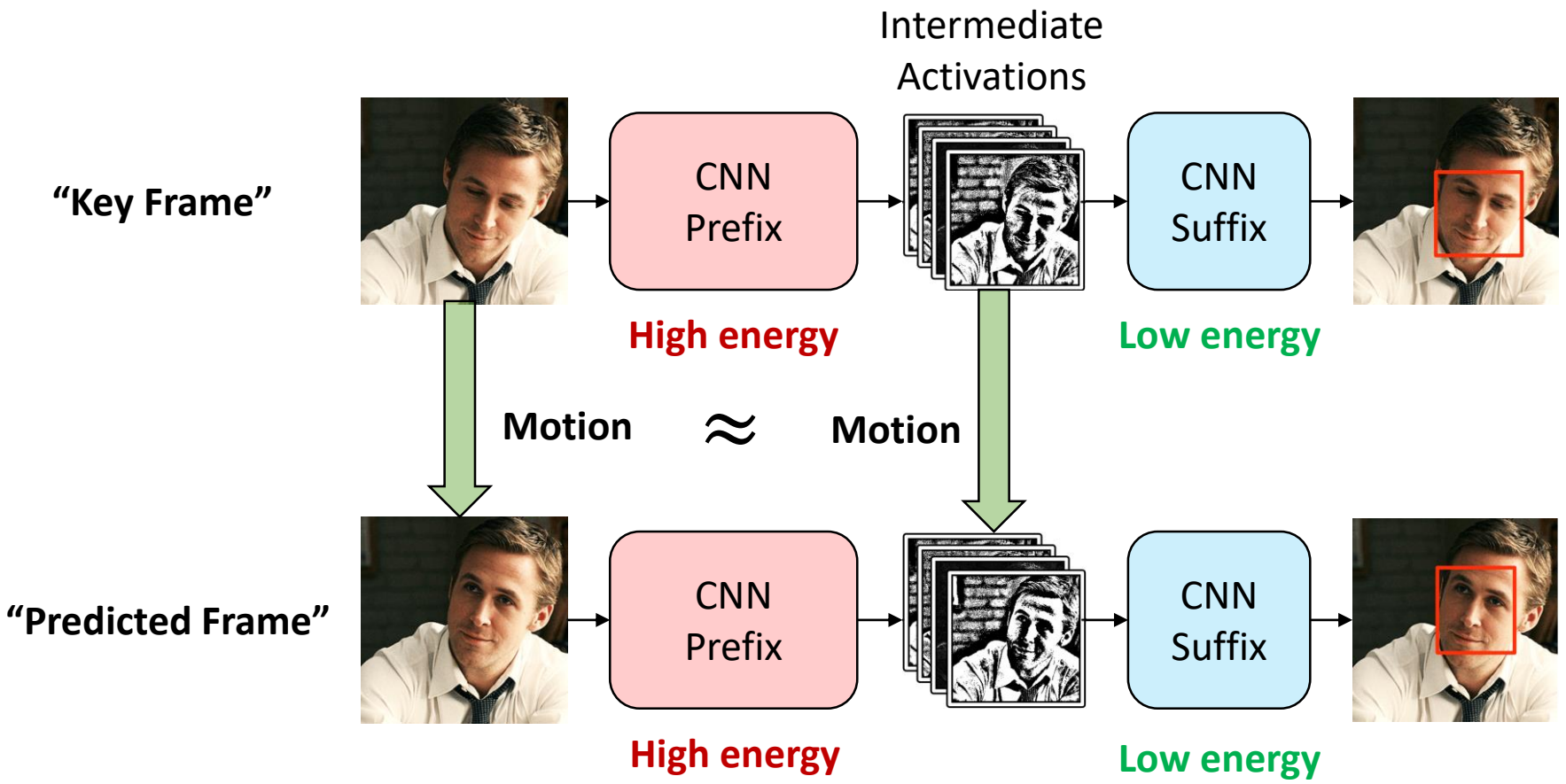
Image Captioning

Common Structure in CNNs



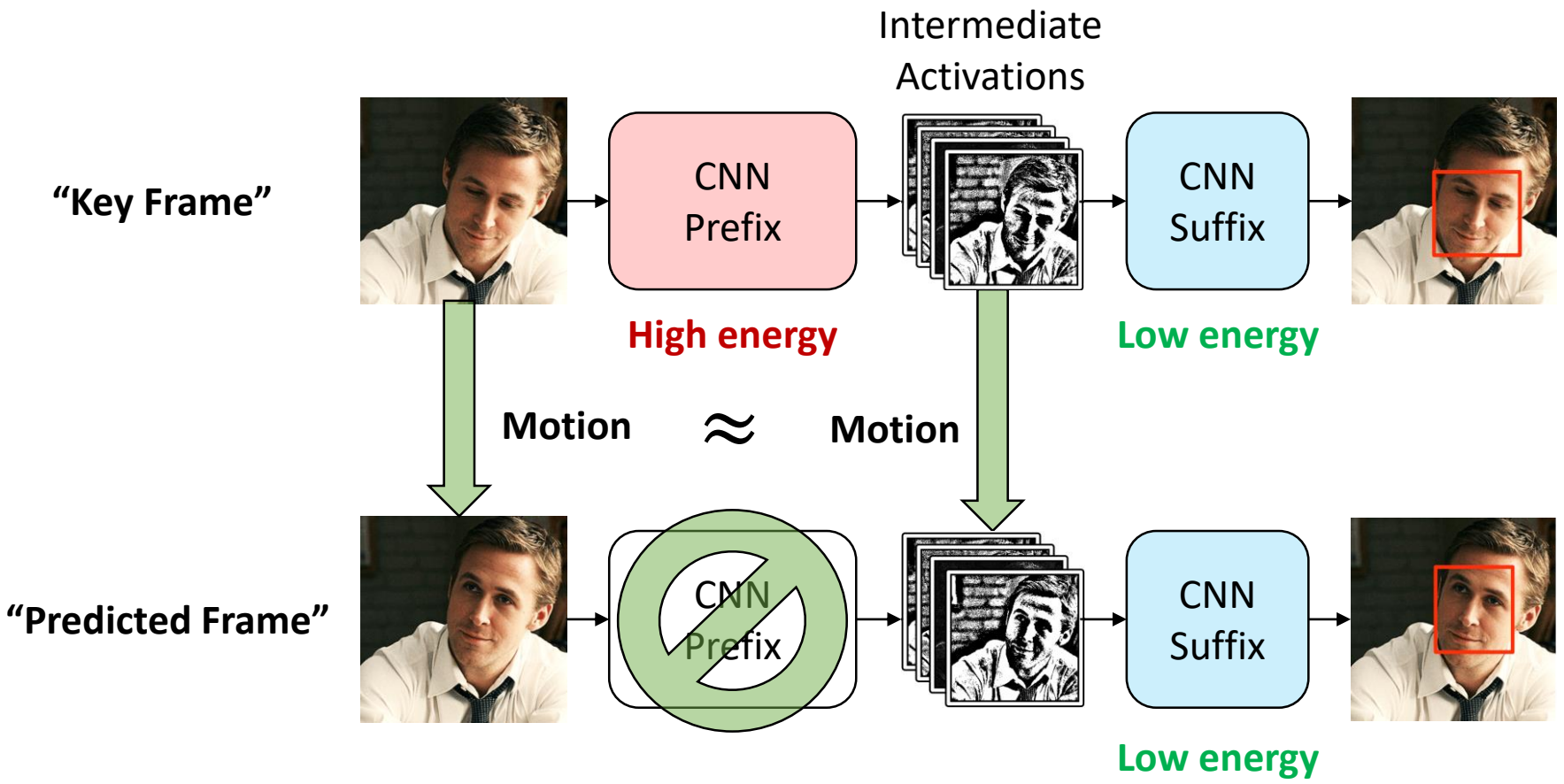
#MakeRyanGoslingTheNewLenna

Common Structure in CNNs



#MakeRyanGoslingTheNewLenna

Common Structure in CNNs



#MakeRyanGoslingTheNewLenna

Talk Overview

Background

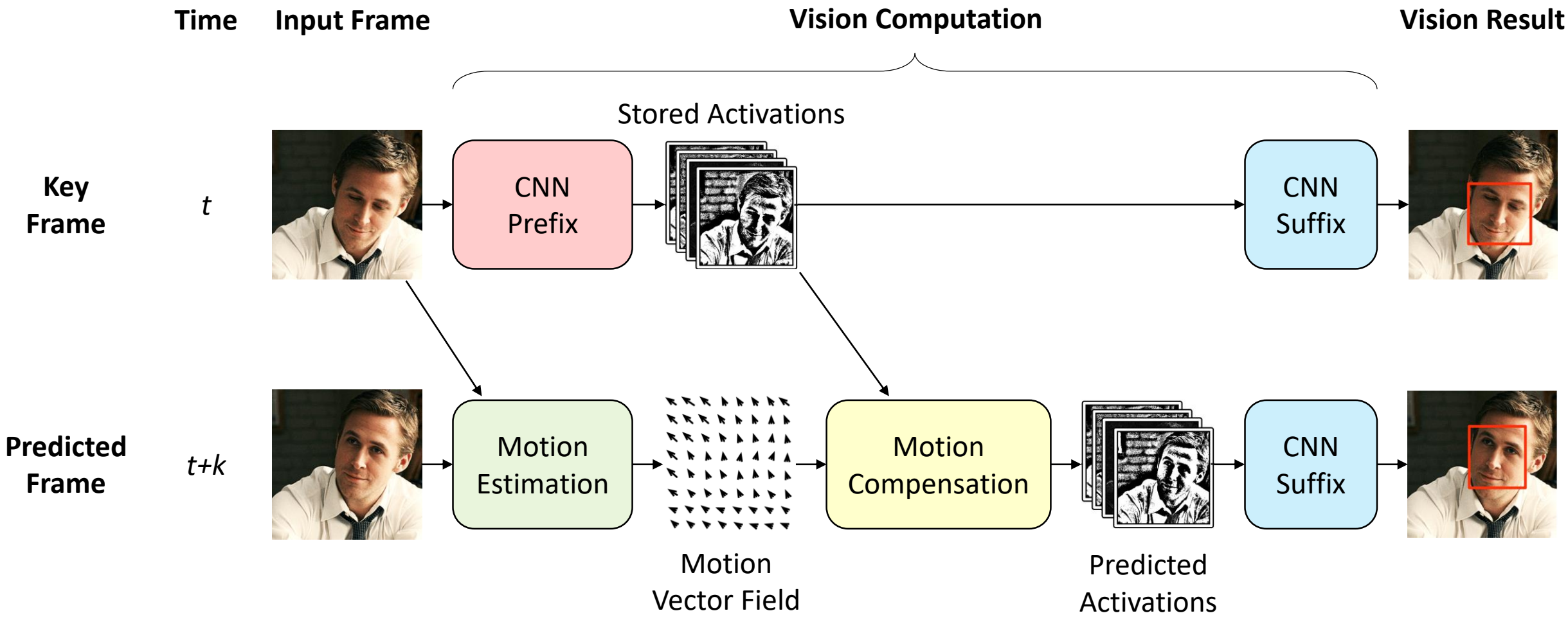
Algorithm

Hardware

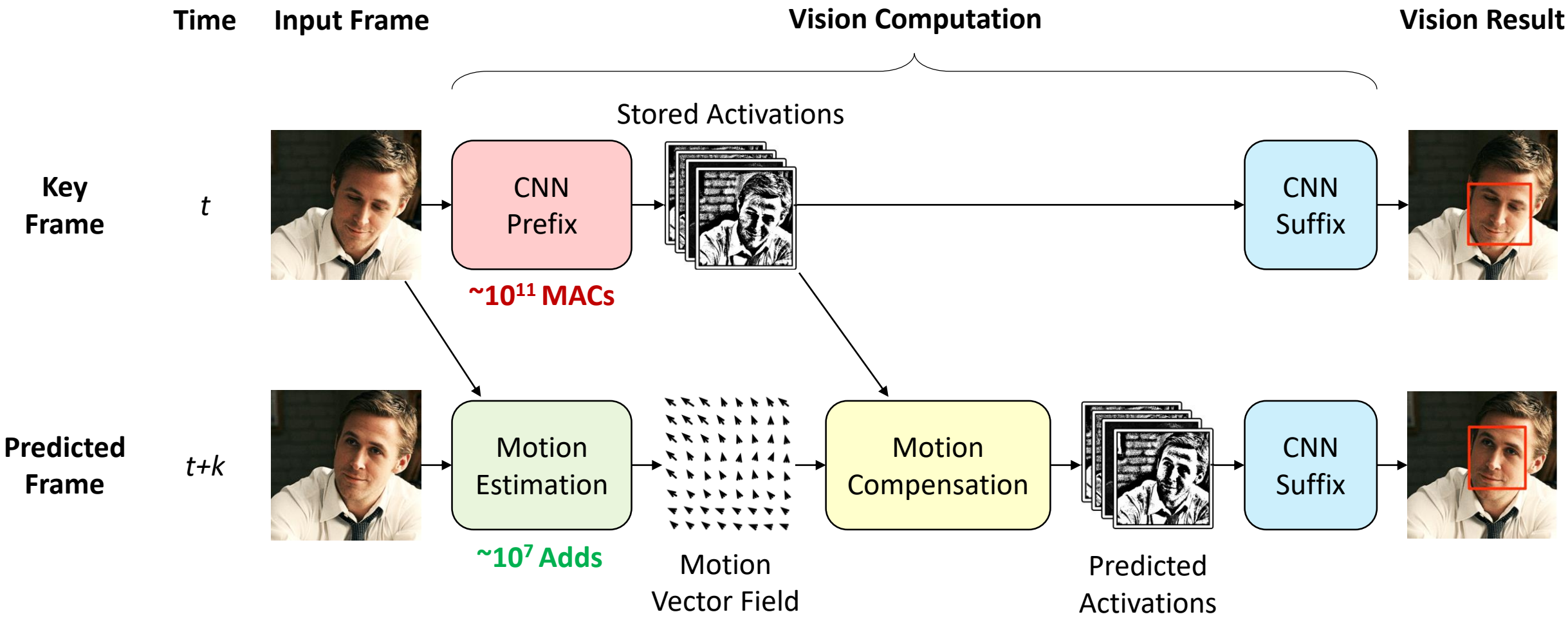
Evaluation

Conclusion

Activation Motion Compensation (AMC)

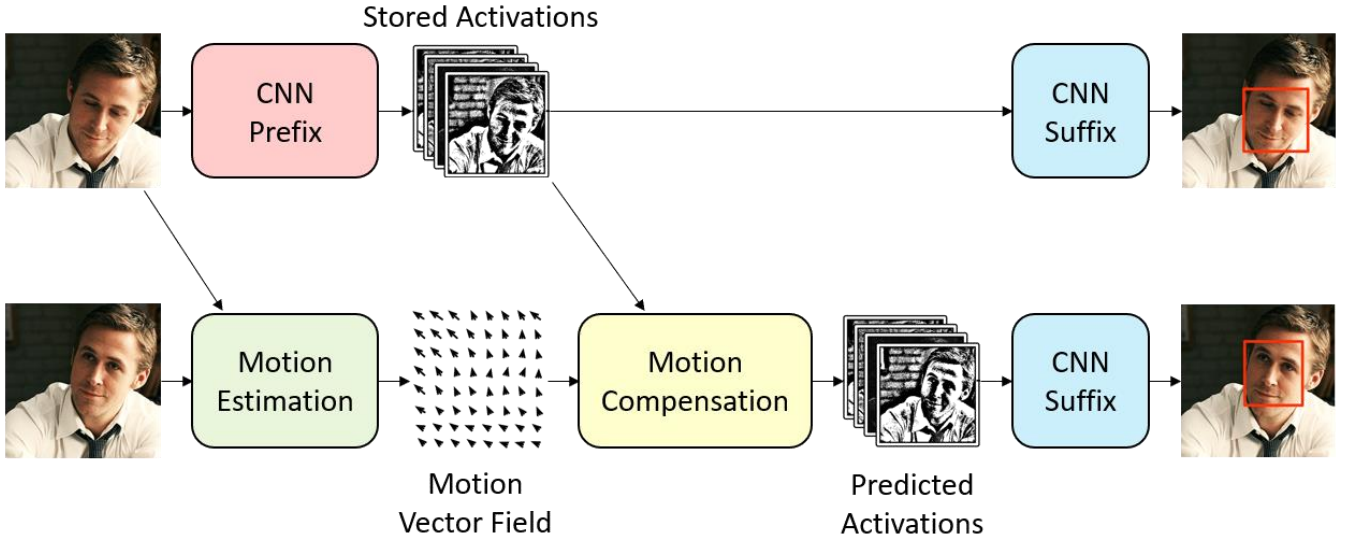


Activation Motion Compensation (AMC)



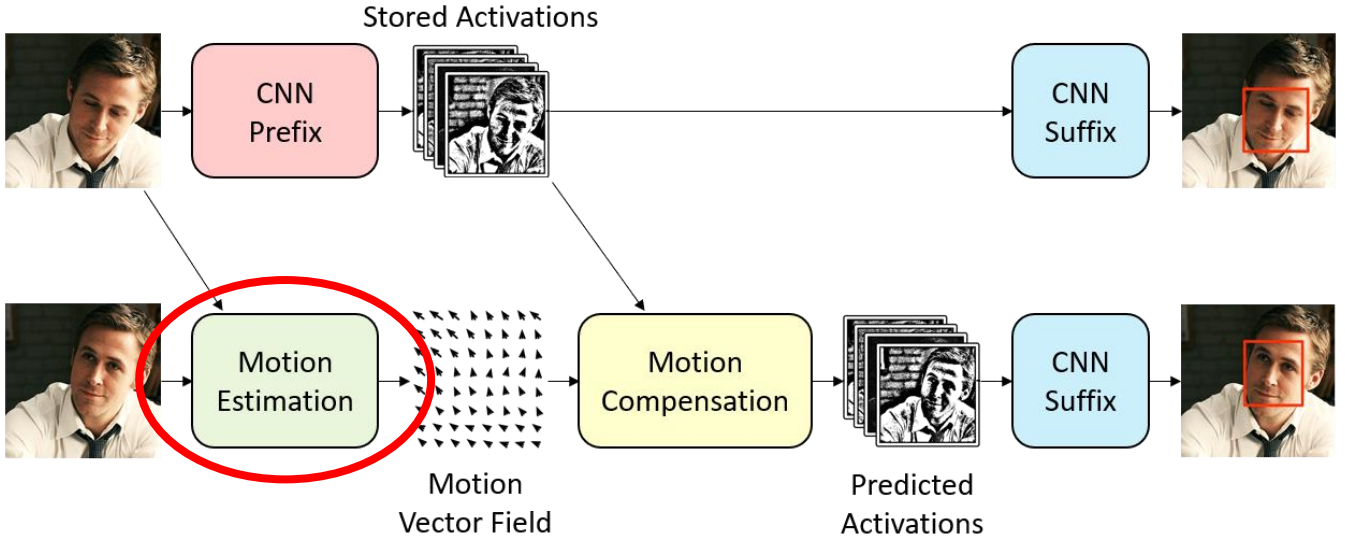
AMC Design Decisions

- How to perform motion estimation?
- How to perform motion compensation?
- Which frames are key frames?



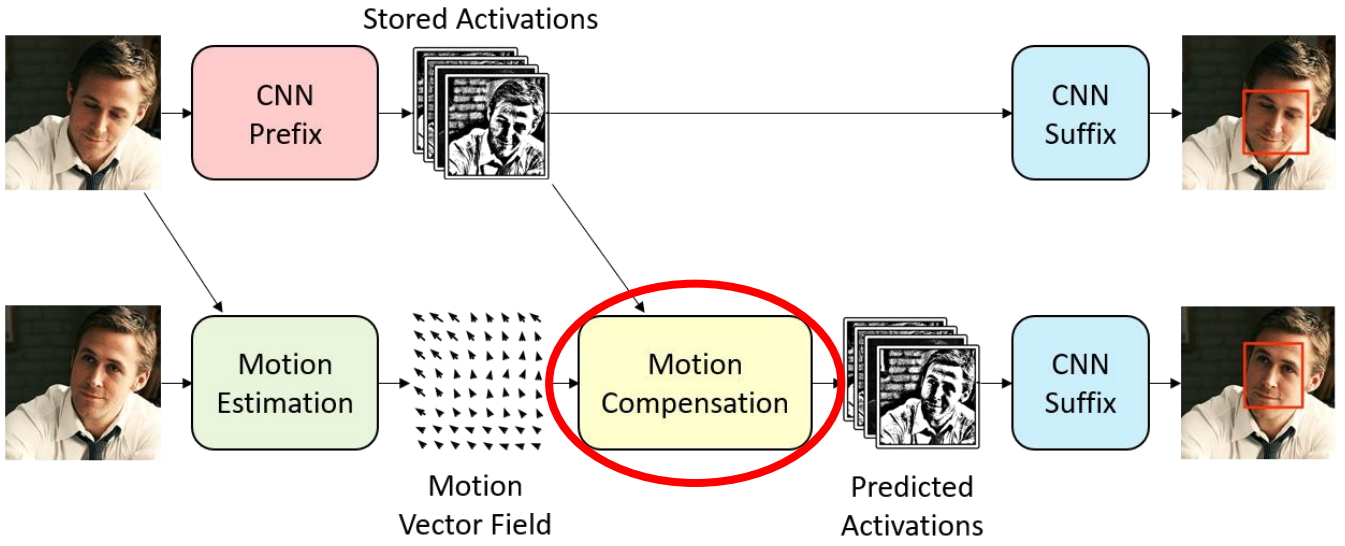
AMC Design Decisions

- **How to perform motion estimation?**
- How to perform motion compensation?
- Which frames are key frames?



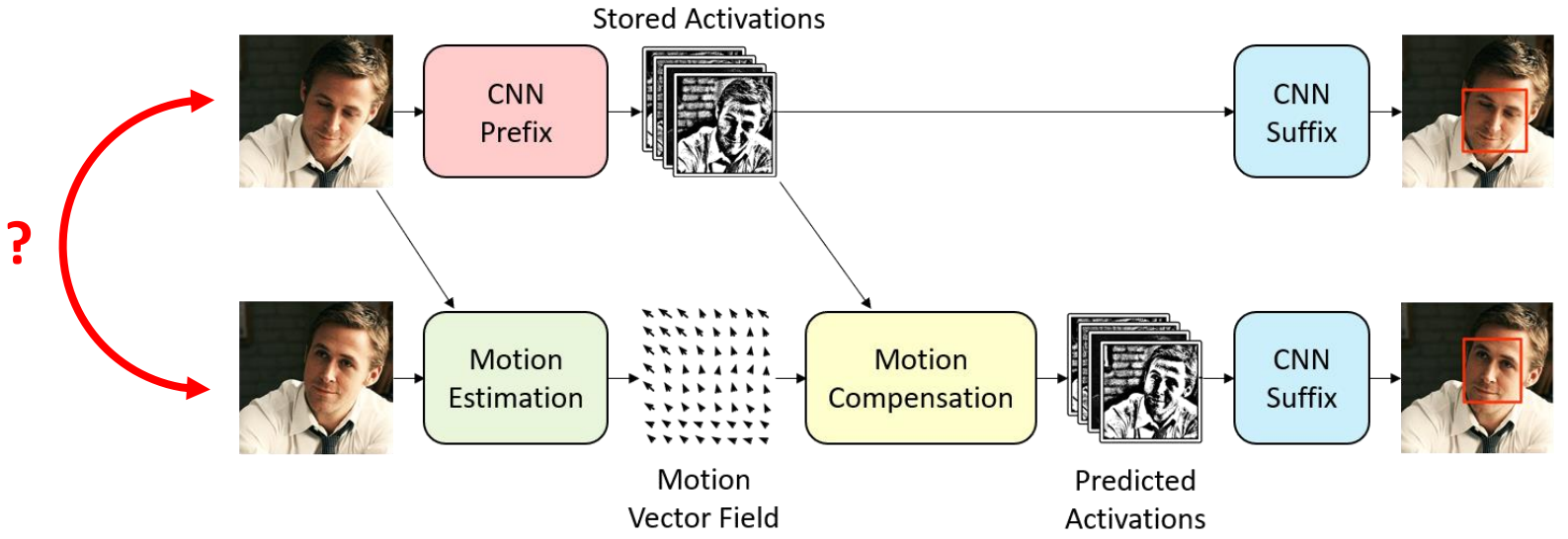
AMC Design Decisions

- How to perform motion estimation?
- **How to perform motion compensation?**
- Which frames are key frames?



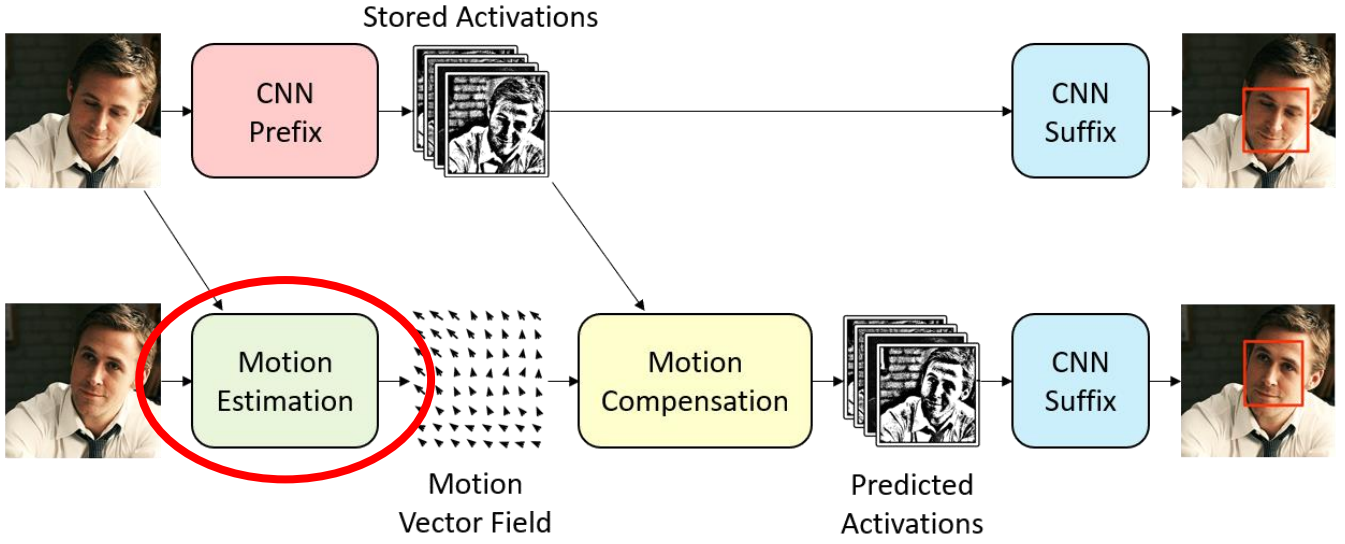
AMC Design Decisions

- How to perform motion estimation?
- How to perform motion compensation?
- **Which frames are key frames?**



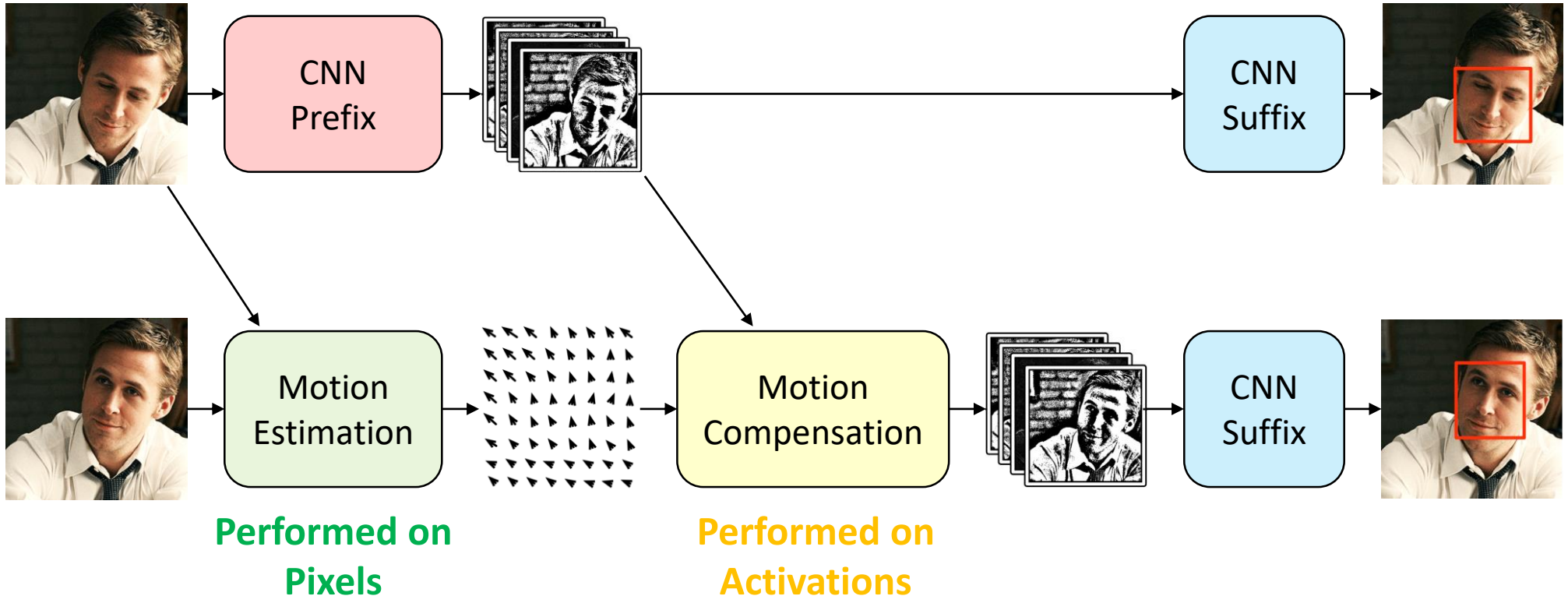
AMC Design Decisions

- **How to perform motion estimation?**
- How to perform motion compensation?
- Which frames are key frames?



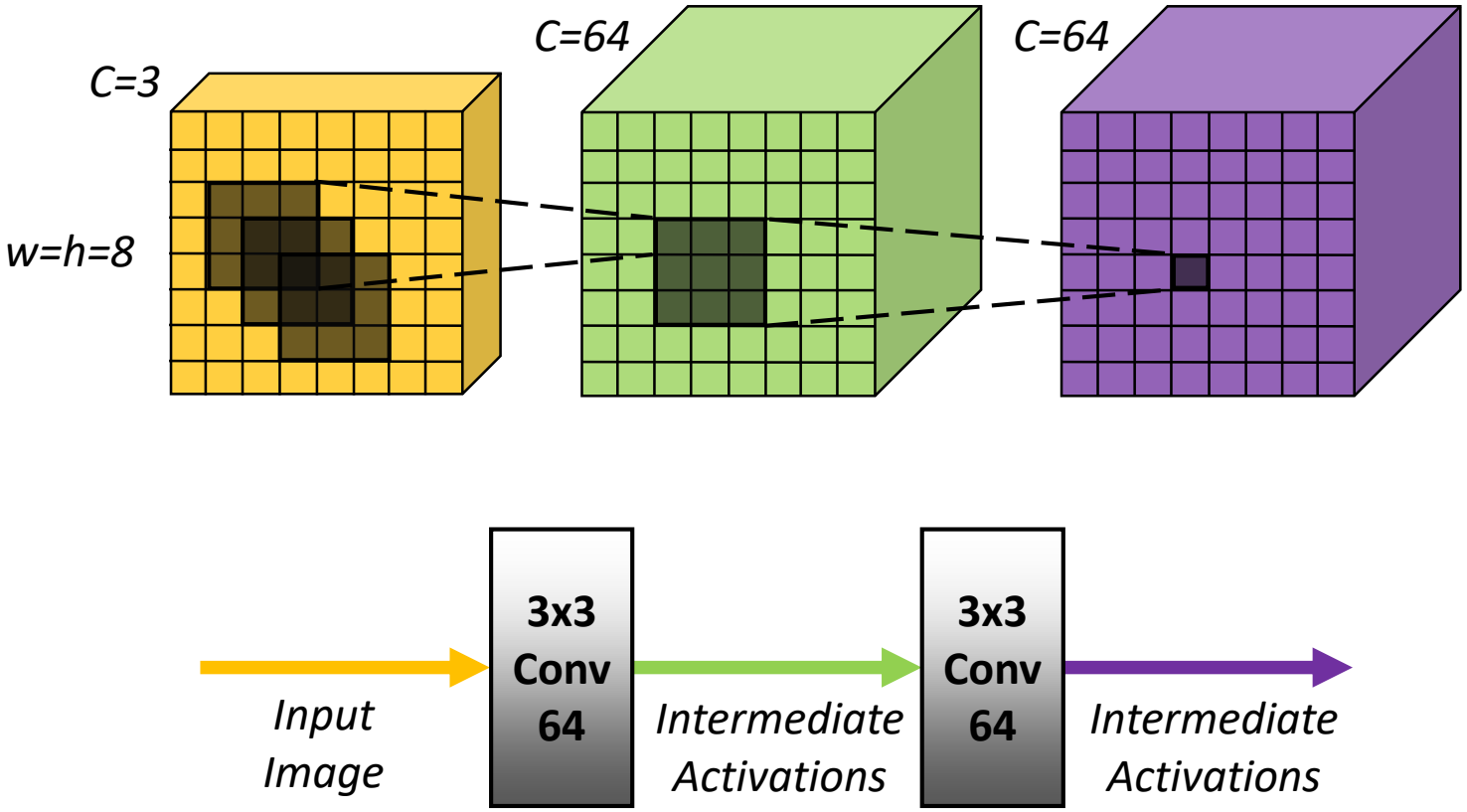
Motion Estimation

- We need to estimate the motion of **activations** by using **pixels**…

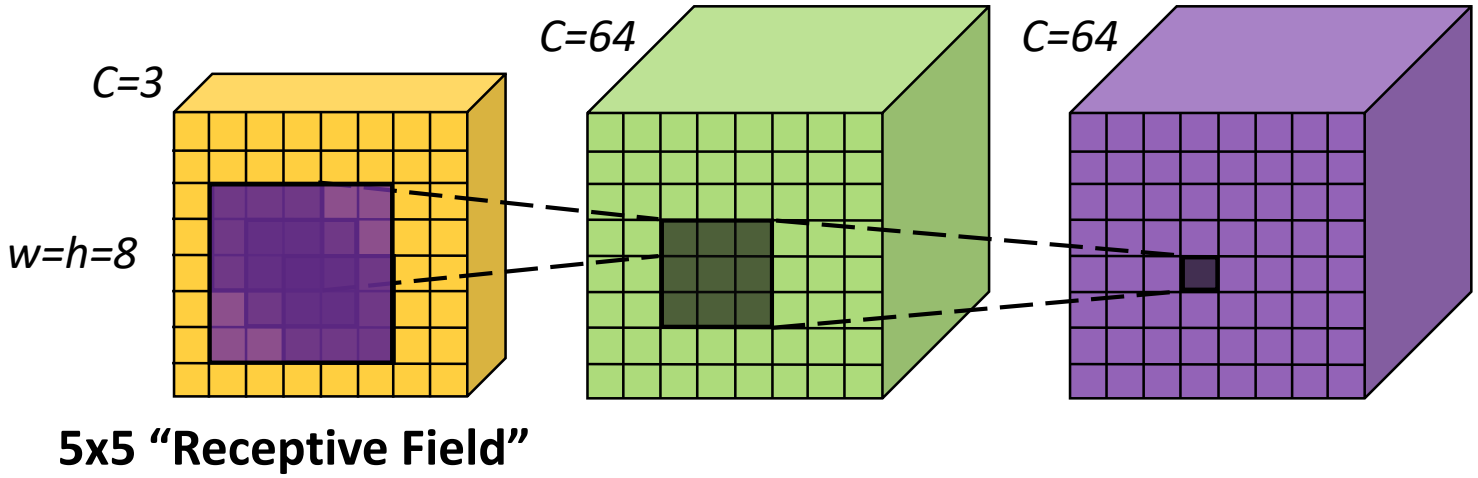


Pixels to Activations

Pixels to Activations: Receptive Fields

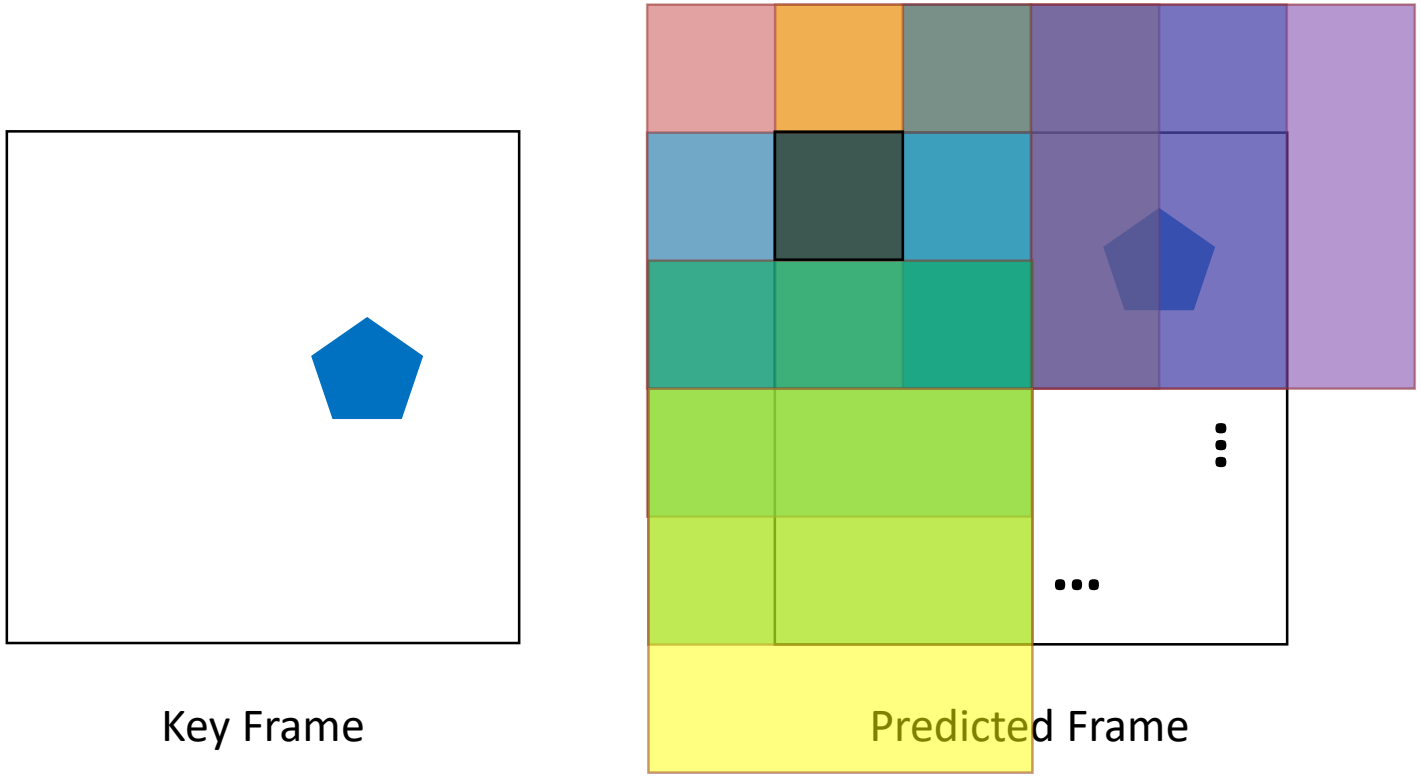


Pixels to Activations: Receptive Fields

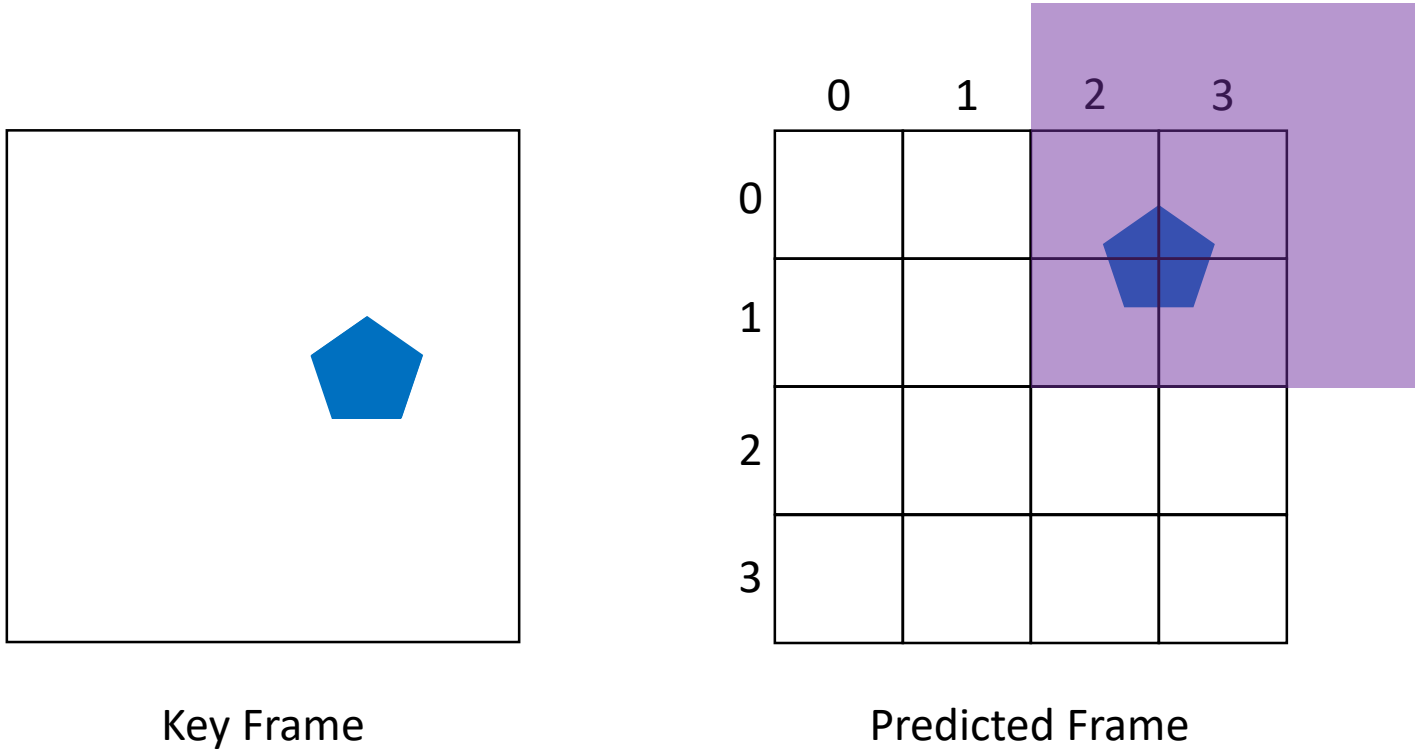


- Estimate motion of **activations** by estimating motion of **receptive fields**

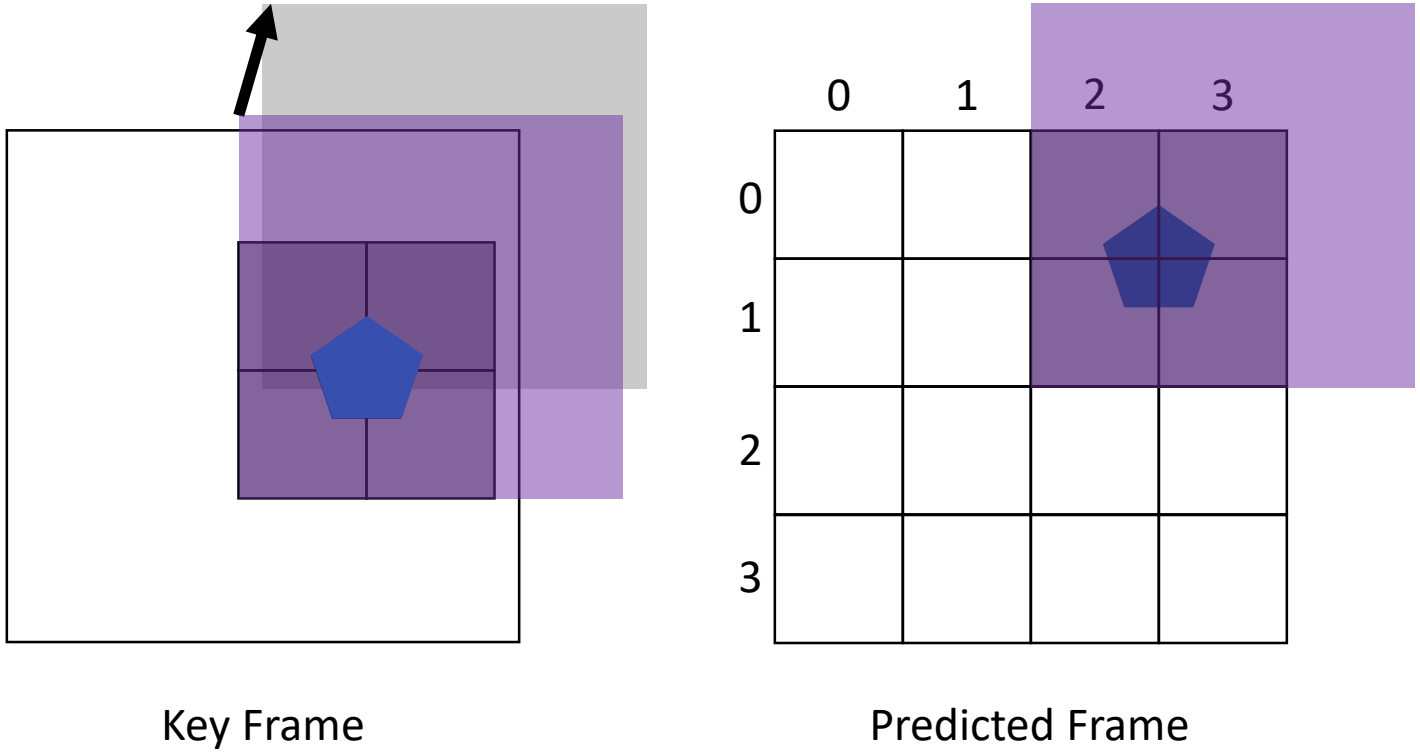
Receptive Field Block Motion Estimation *(RFBME)*



Receptive Field Block Motion Estimation *(RFBME)*

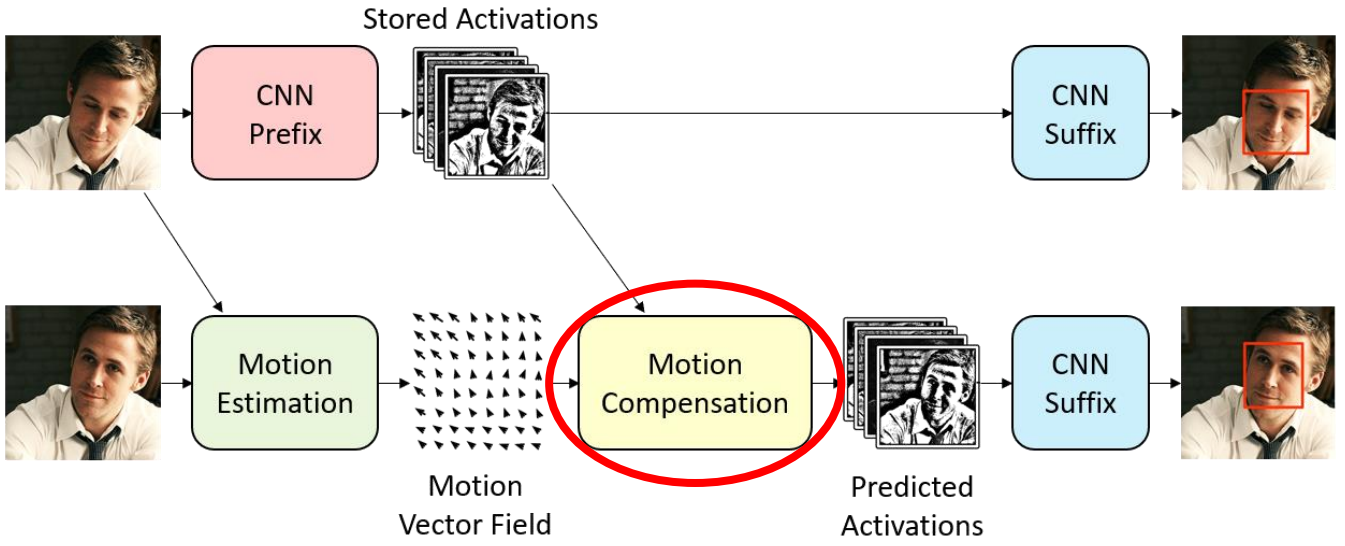


Receptive Field Block Motion Estimation *(RFBME)*



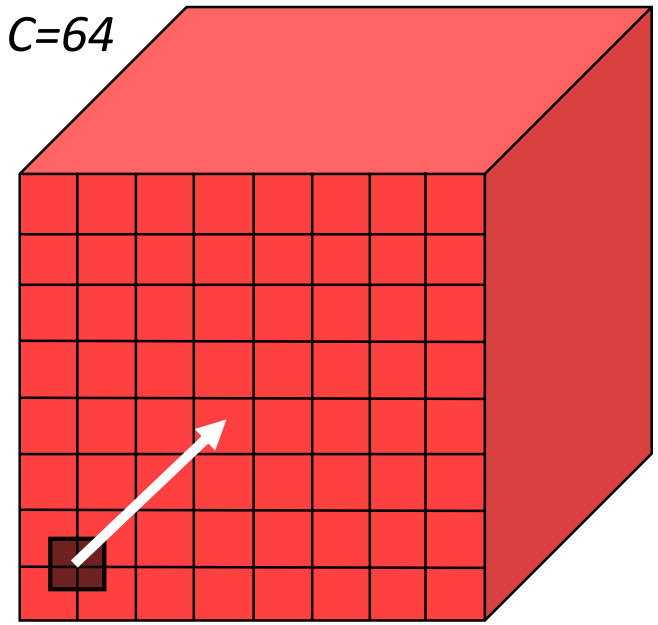
AMC Design Decisions

- How to perform motion estimation?
- **How to perform motion compensation?**
- Which frames are key frames?

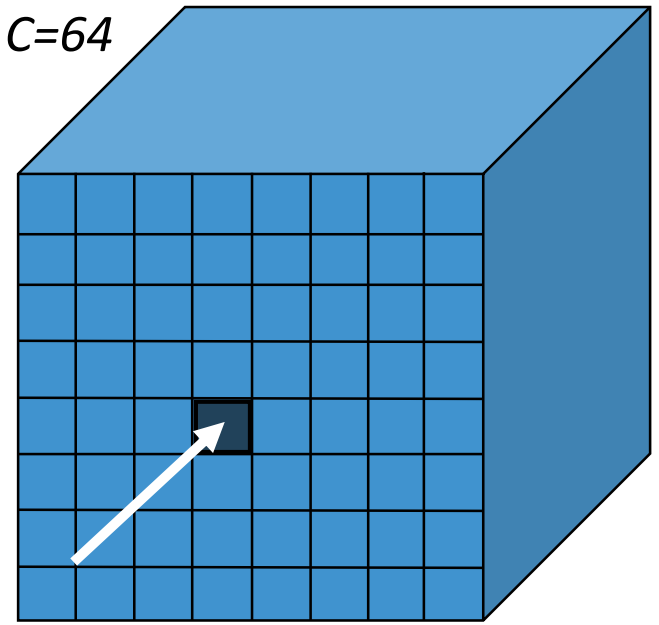


Motion Compensation

Vector:
X = 2.5
Y = 2.5



Stored Activations

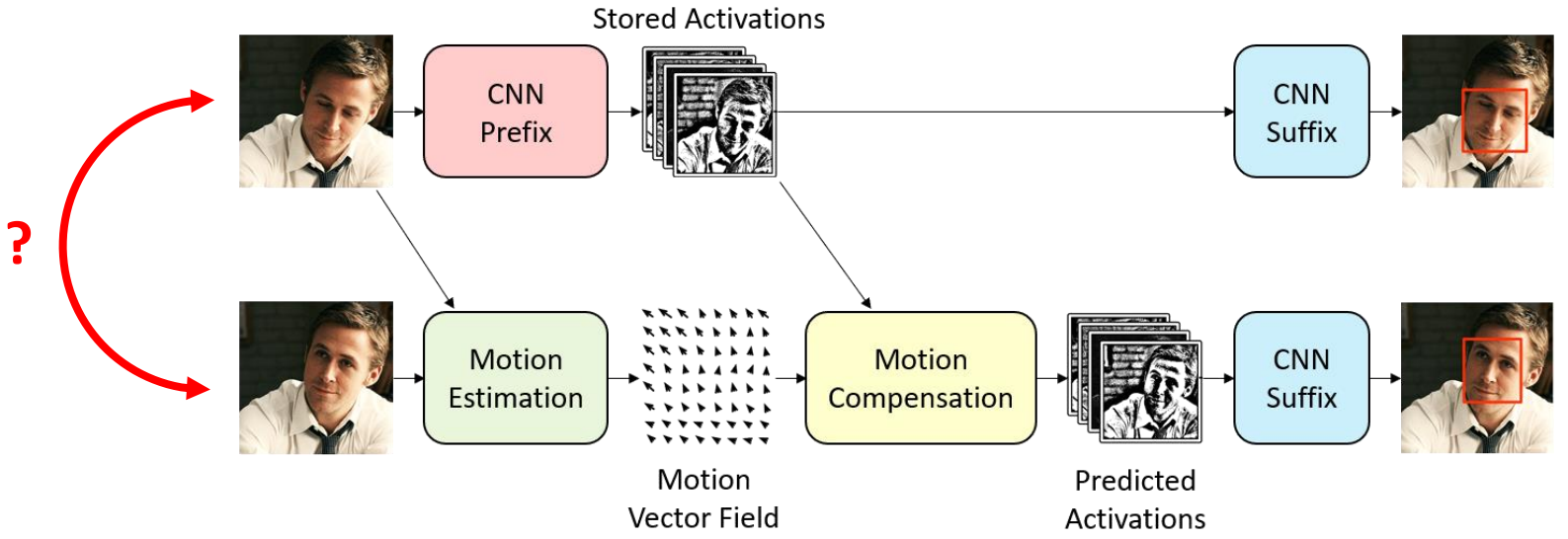


Predicted Activations

- Subtract the vector to index into the stored activations
- Interpolate when necessary

AMC Design Decisions

- How to perform motion estimation?
- How to perform motion compensation?
- **Which frames are key frames?**

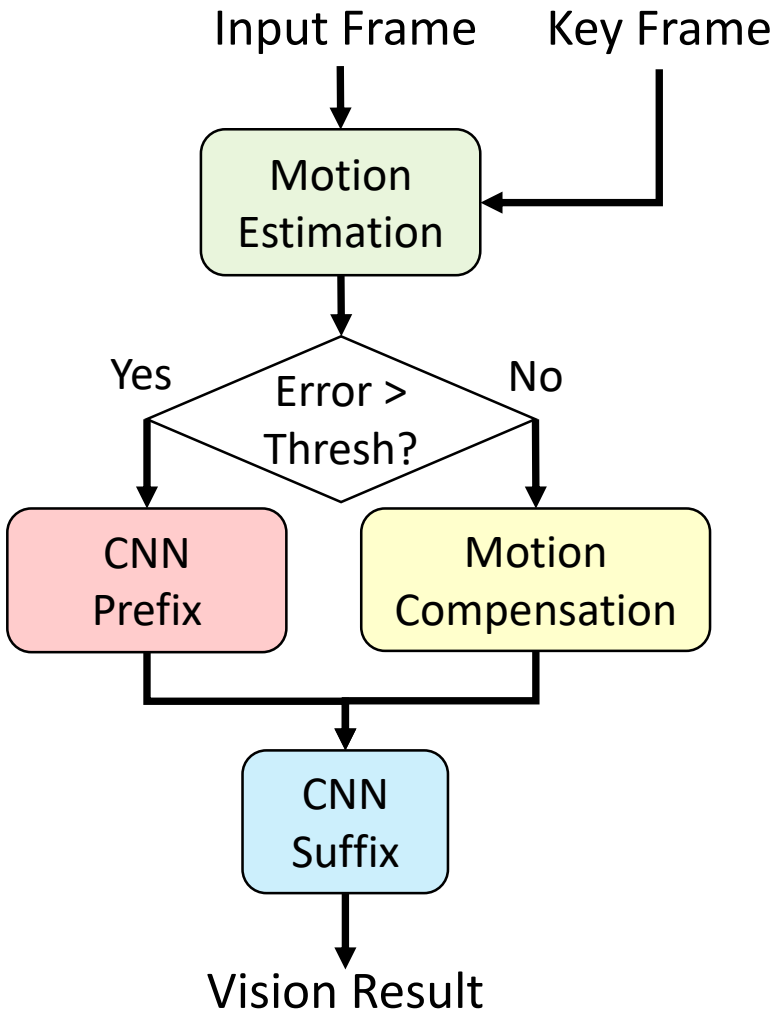


When to Compute Key Frame?

- System needs a new key frame when motion estimation fails:
  - De-occlusion
  - New objects
  - Rotation/scaling
  - Lighting changes

When to Compute Key Frame?

- System needs a new key frame when motion estimation fails:
  - De-occlusion
  - New objects
  - Rotation/scaling
  - Lighting changes
- So, compute key frame when *RFBME error exceeds set threshold*



Talk Overview

Background

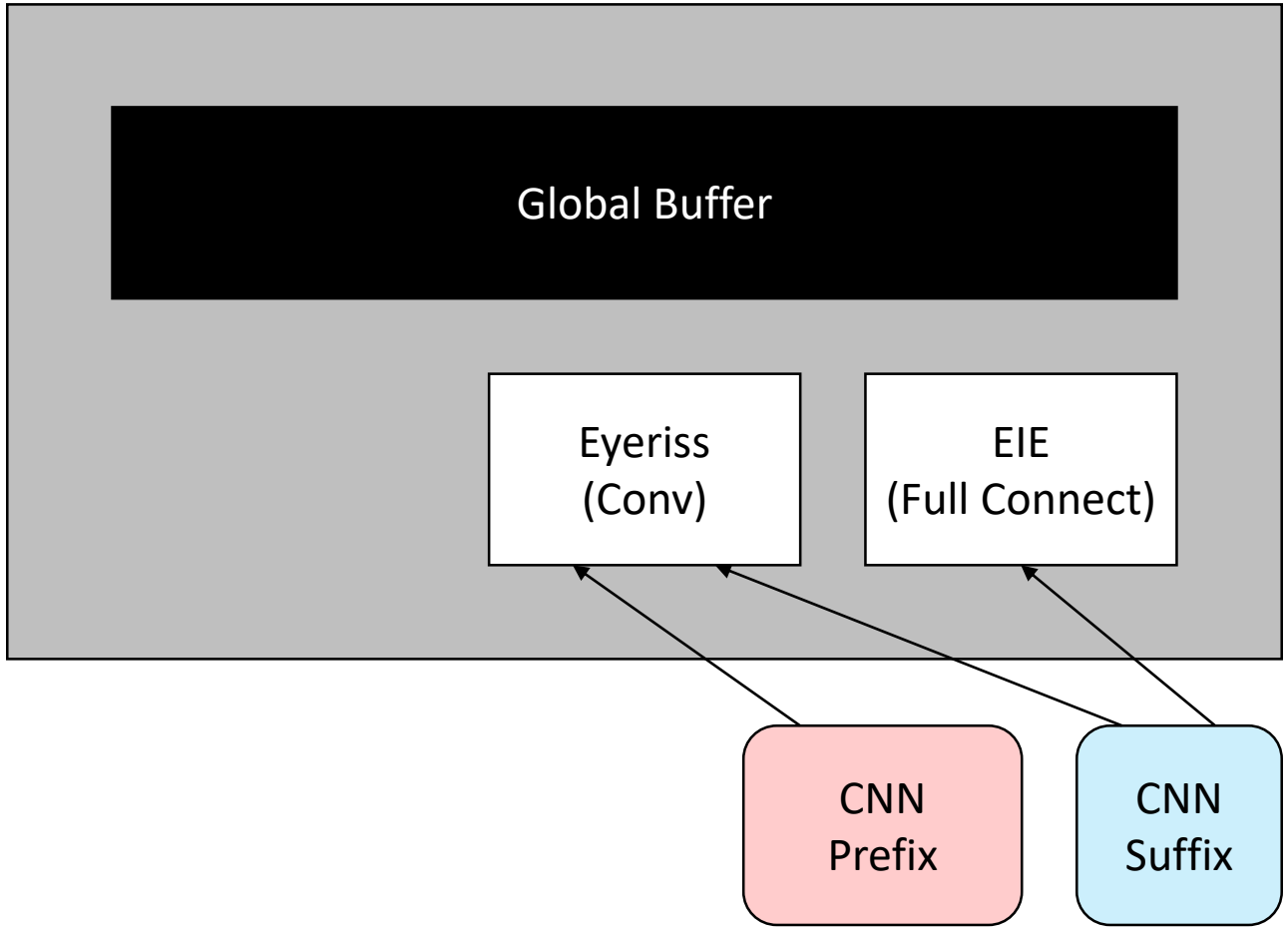
Algorithm

Hardware

Evaluation

Conclusion

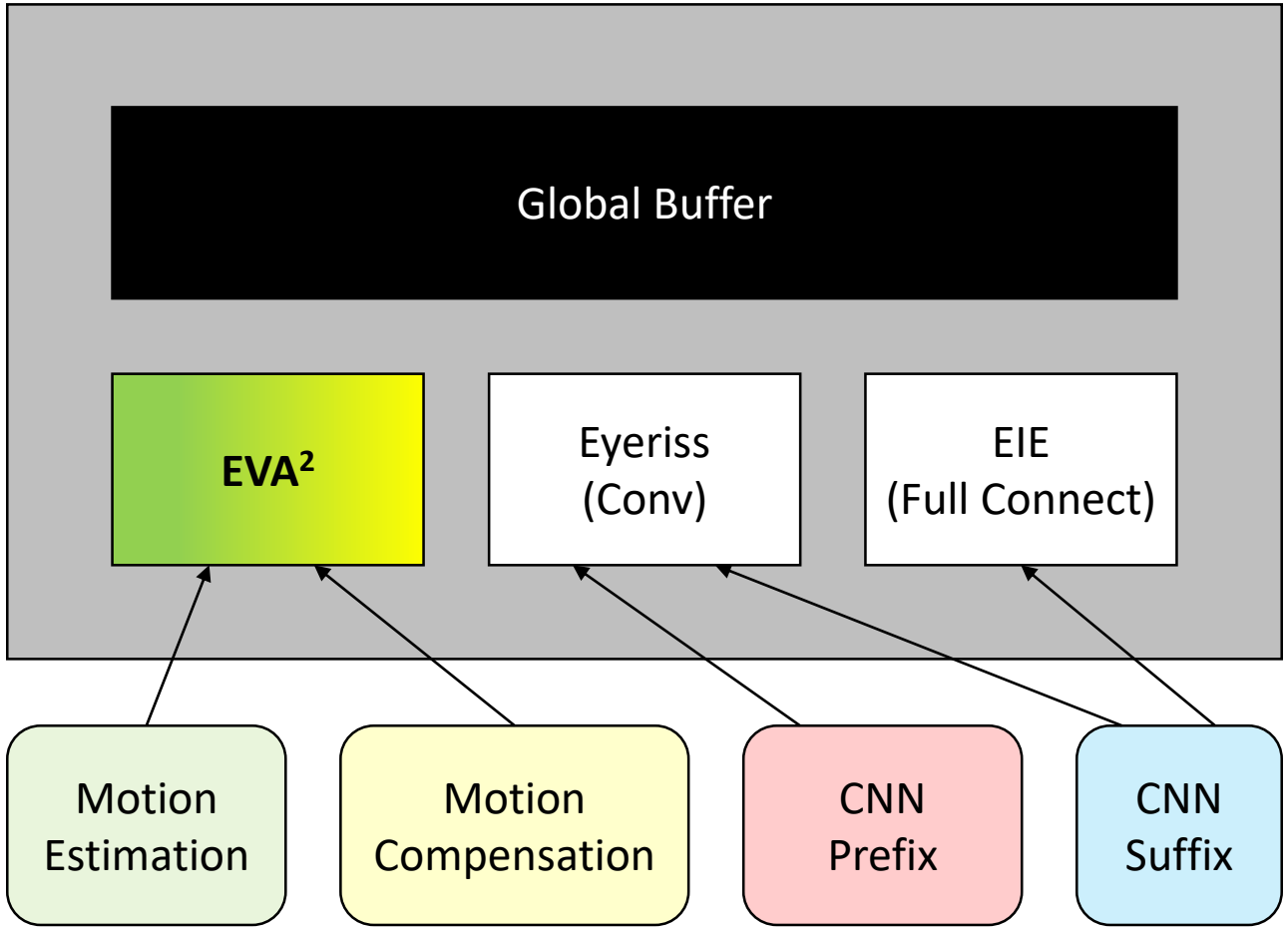
Embedded Vision Accelerator



Y.-H. Chen, T. Krishna, J. S. Emer, and V. Sze,
"Eyeriss: An energy-efficient reconfigurable accelerator for deep convolutional neural networks,"

S. Han, X. Liu, H. Mao, J. Pu, A. Pedram, M. A. Horowitz, and W. J. Dally,
"EIE: Efficient inference engine on compressed deep neural network,"

Embedded Vision Accelerator Accelerator (EVA^2)

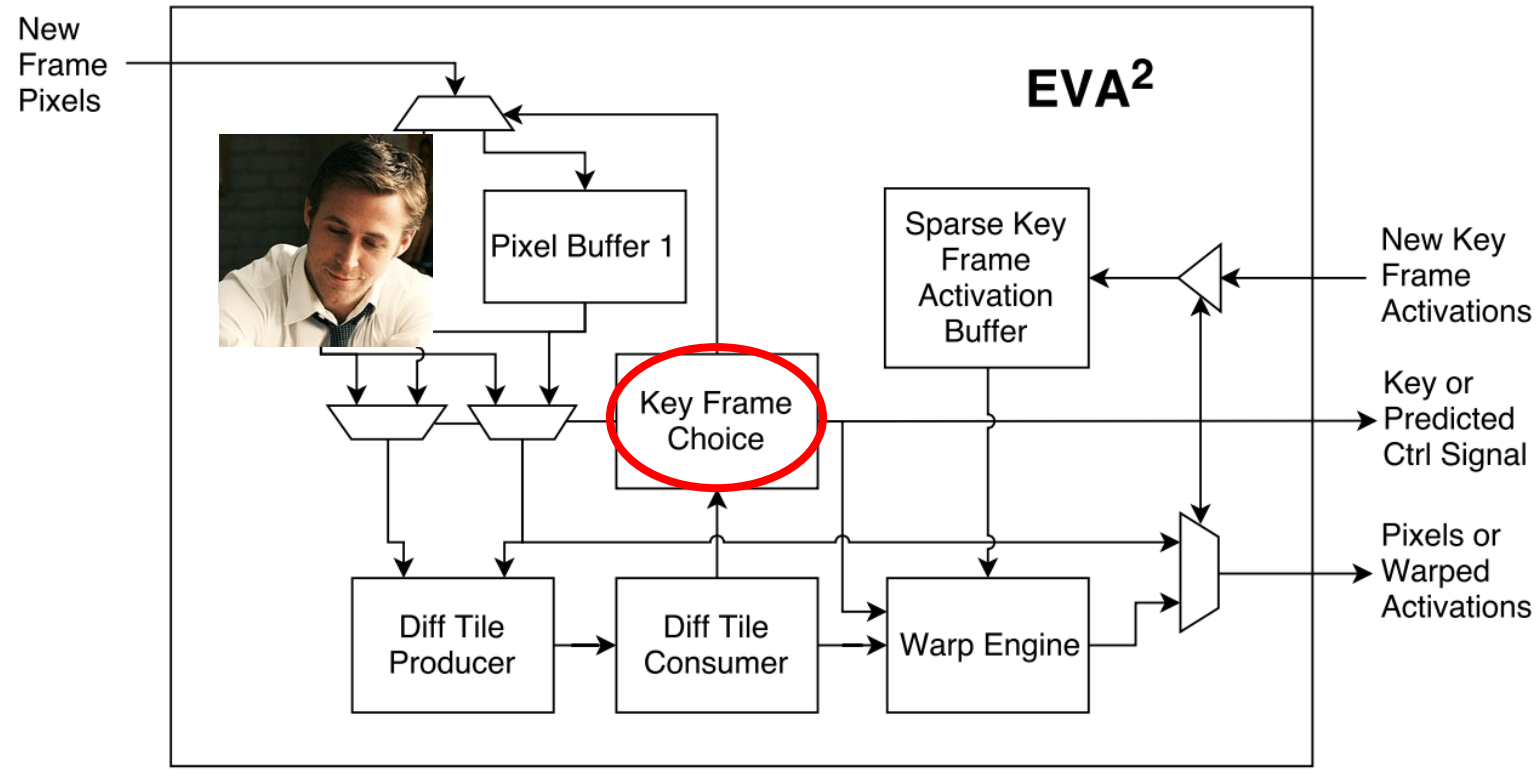


Y.-H. Chen, T. Krishna, J. S. Emer, and V. Sze,
"Eyeriss: An energy-efficient reconfigurable accelerator for deep convolutional neural networks,"

S. Han, X. Liu, H. Mao, J. Pu, A. Pedram, M. A. Horowitz, and W. J. Dally,
"EIE: Efficient inference engine on compressed deep neural network,"

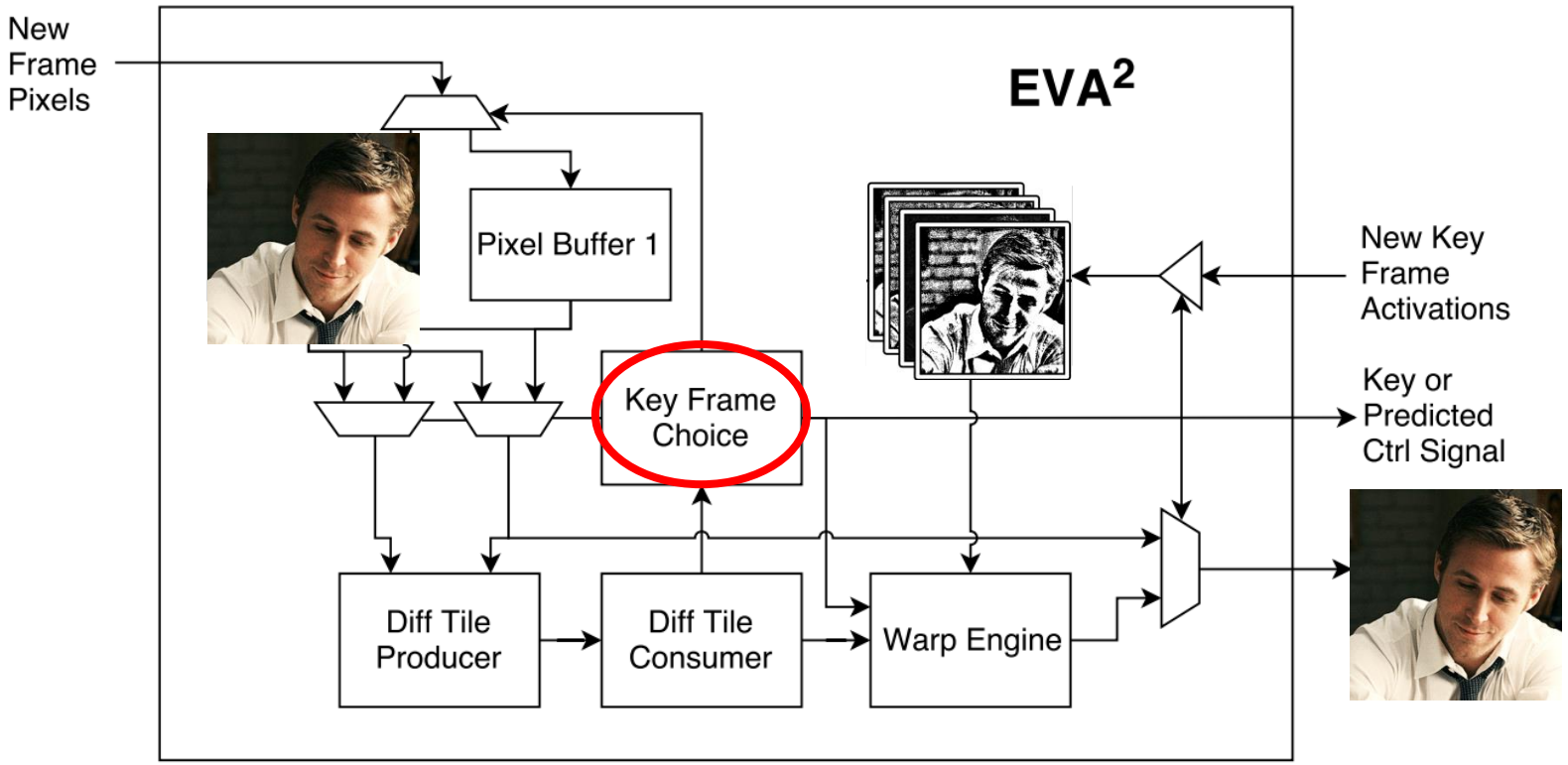
Embedded Vision Accelerator Accelerator (EVA^2)

Frame 0



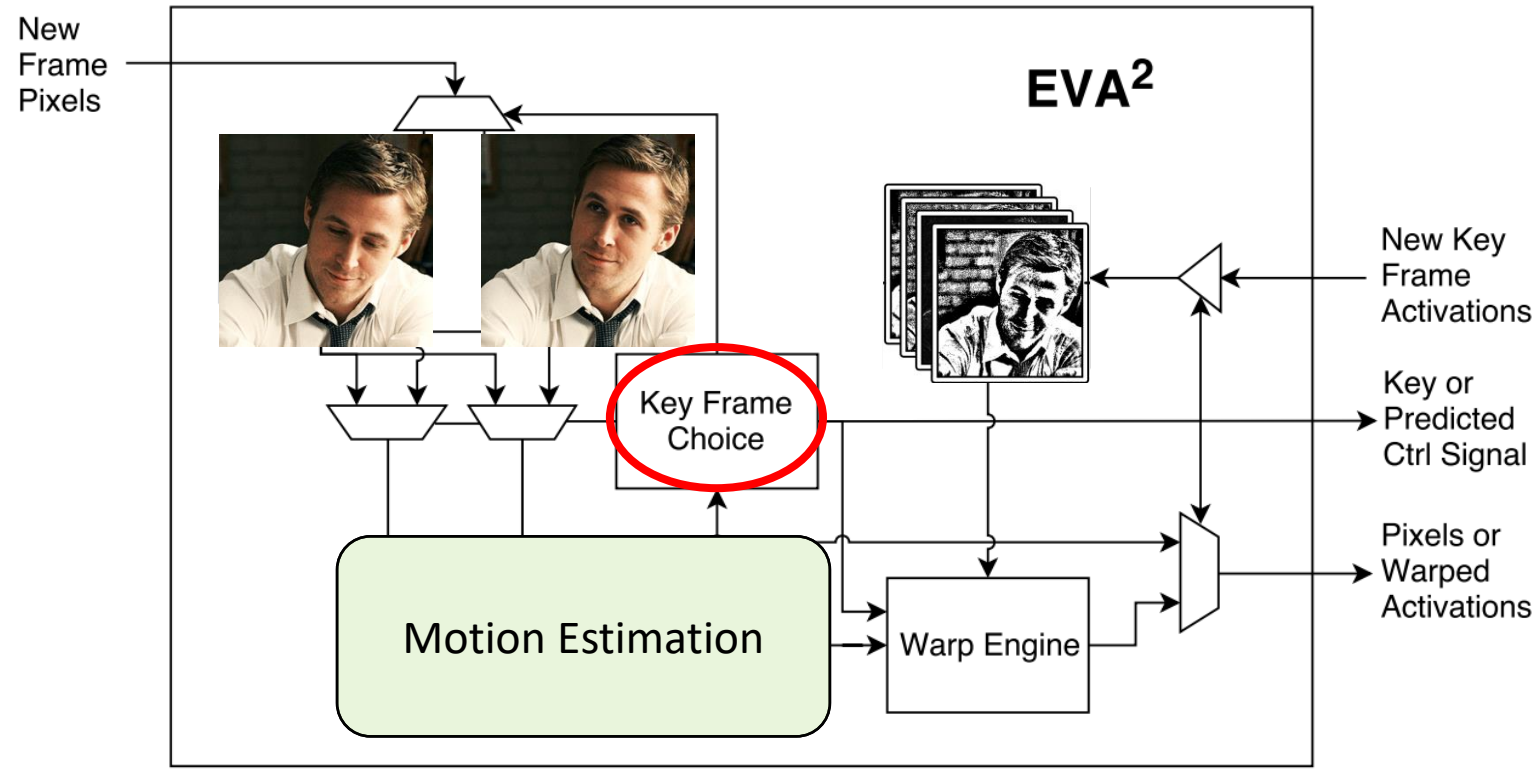
Embedded Vision Accelerator Accelerator (EVA^2)

Frame 0:
Key frame



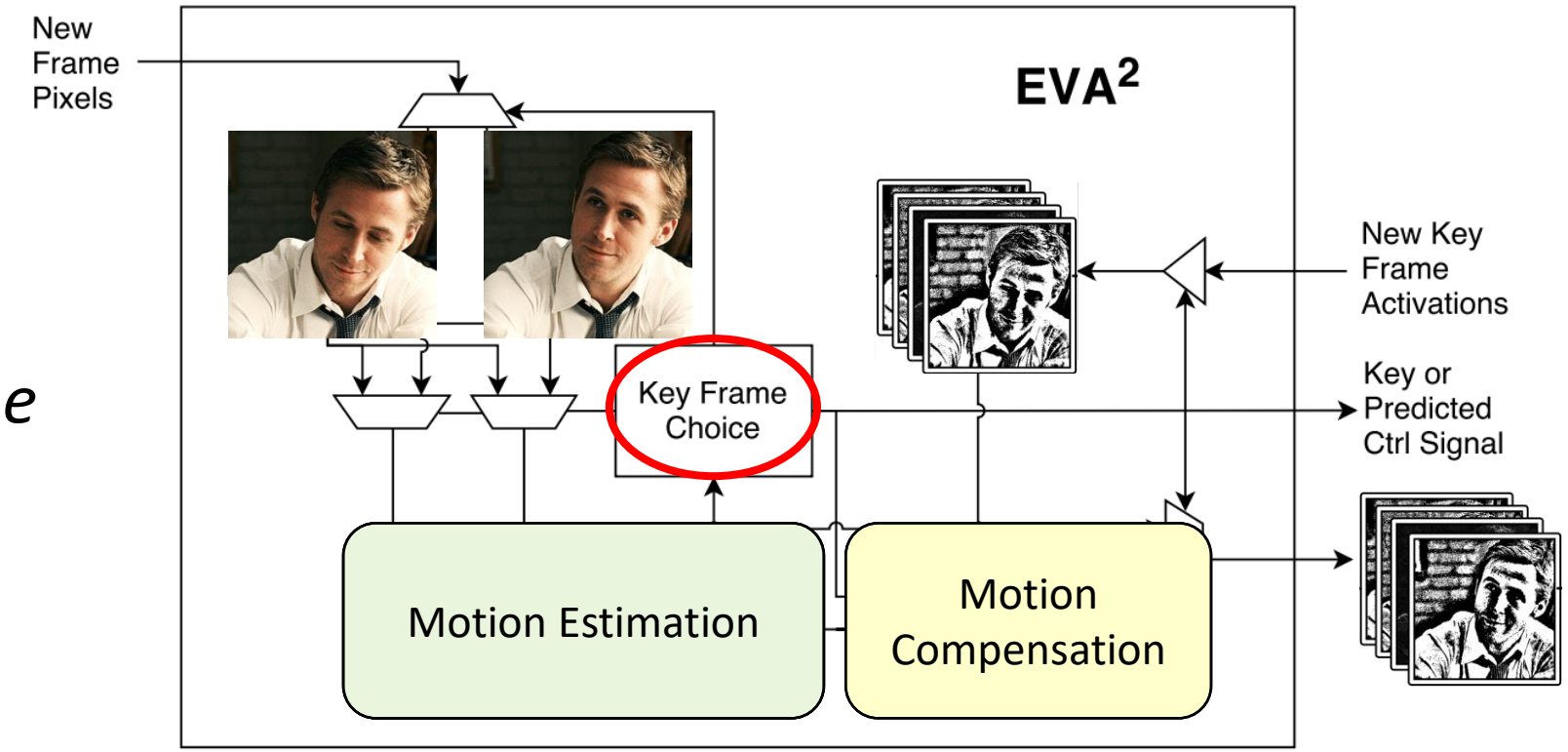
Embedded Vision Accelerator Accelerator (EVA^2)

Frame 1



Embedded Vision Accelerator Accelerator (EVA^2)

Frame 1:
Predicted frame



- EVA^2 leverages sparse techniques to save *80-87% storage and computation*

Talk Overview

Background

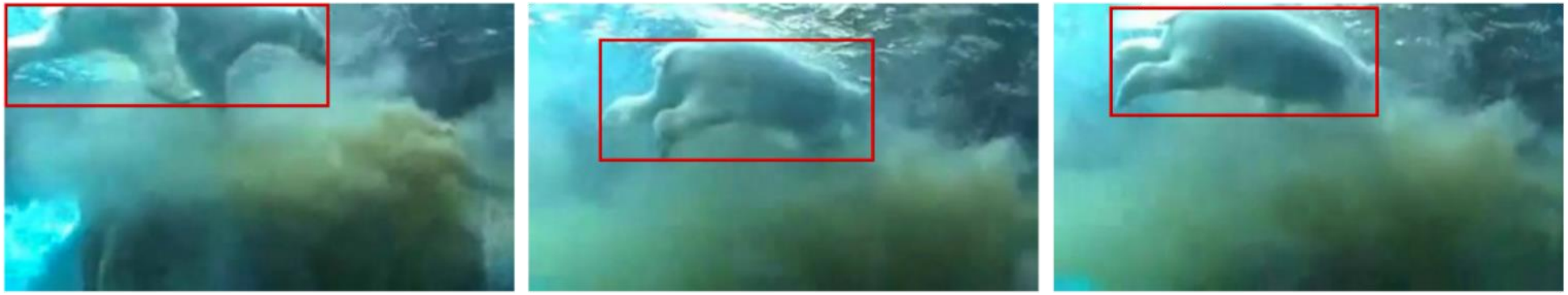
Algorithm

Hardware

Evaluation

Conclusion

Evaluation Details



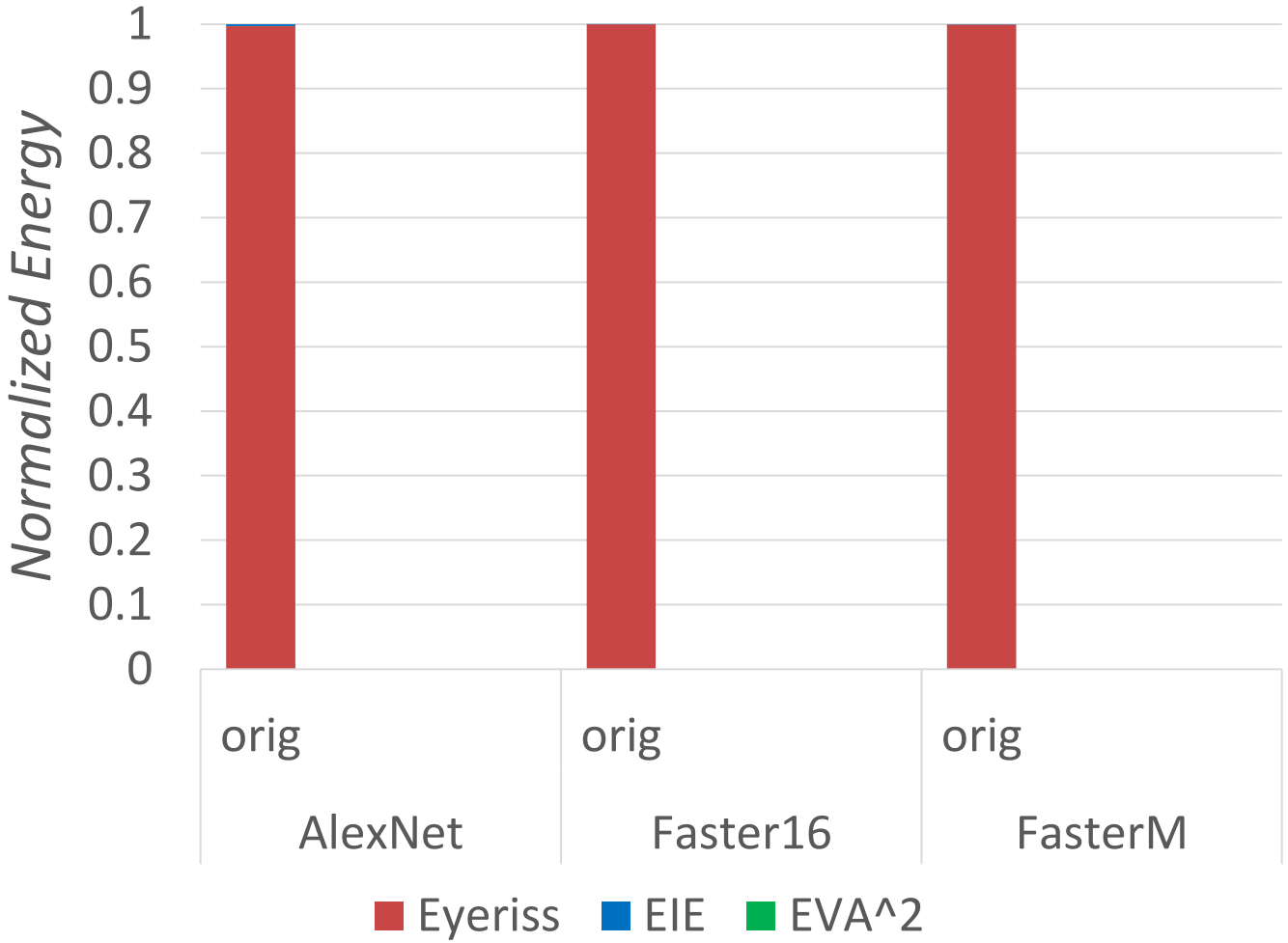
Train/Validation Datasets	YouTube Bounding Box: Object Detection & Classification
Evaluated Networks	AlexNet, Faster R-CNN with VGGM and VGG16
Hardware Baseline	Eyeriss & EIE performance scaled from papers
EVA^2 Implementation	Written in RTL, synthesized with 65nm TSMC

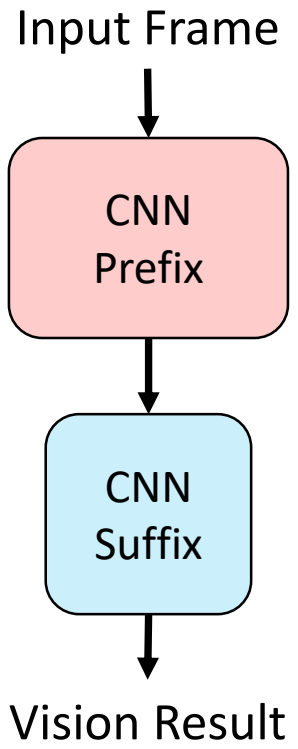
EVA^2 Area Overhead

Total 65nm area: 74mm^2

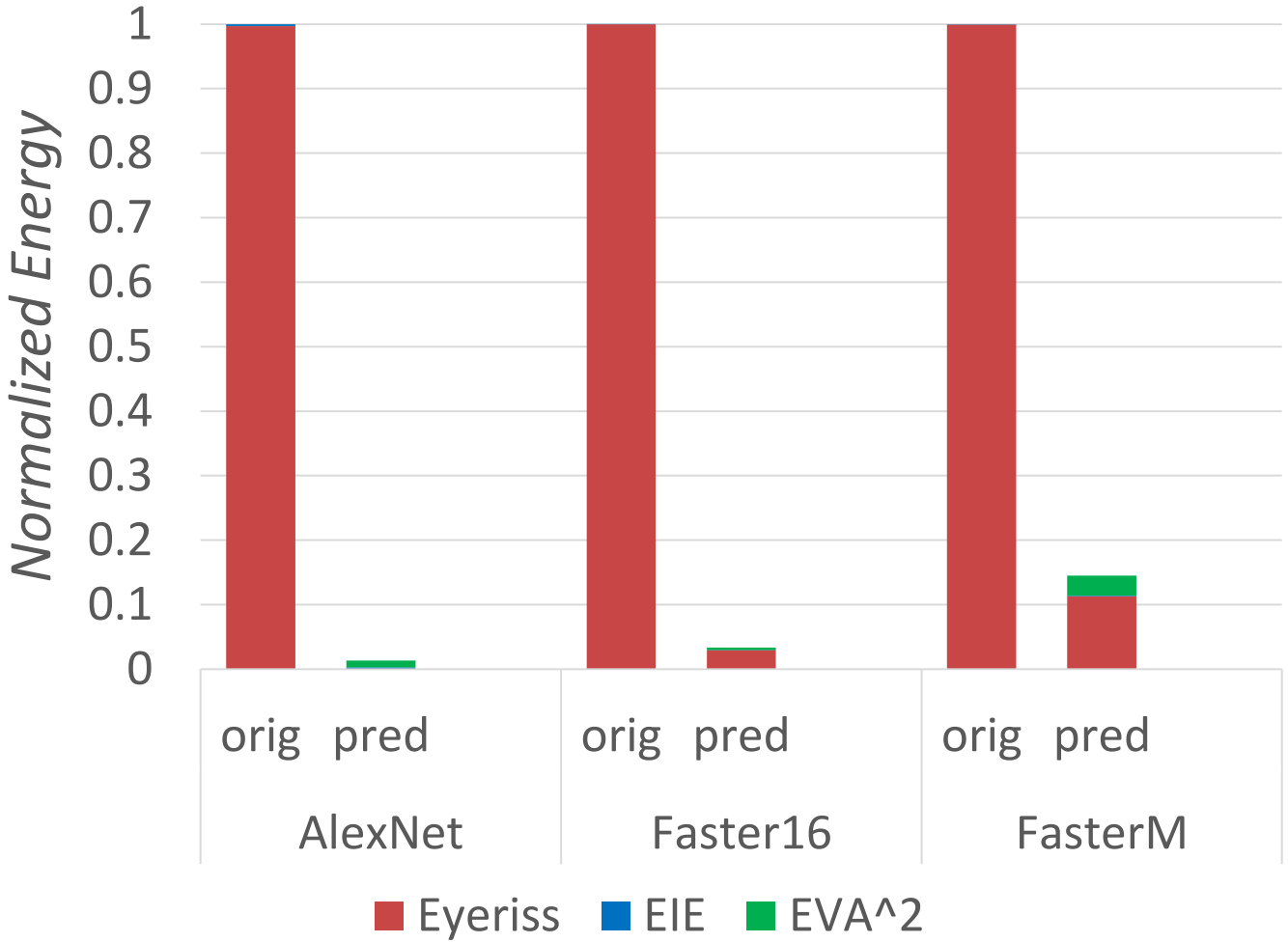
EVA^2 takes up only **3.3%**

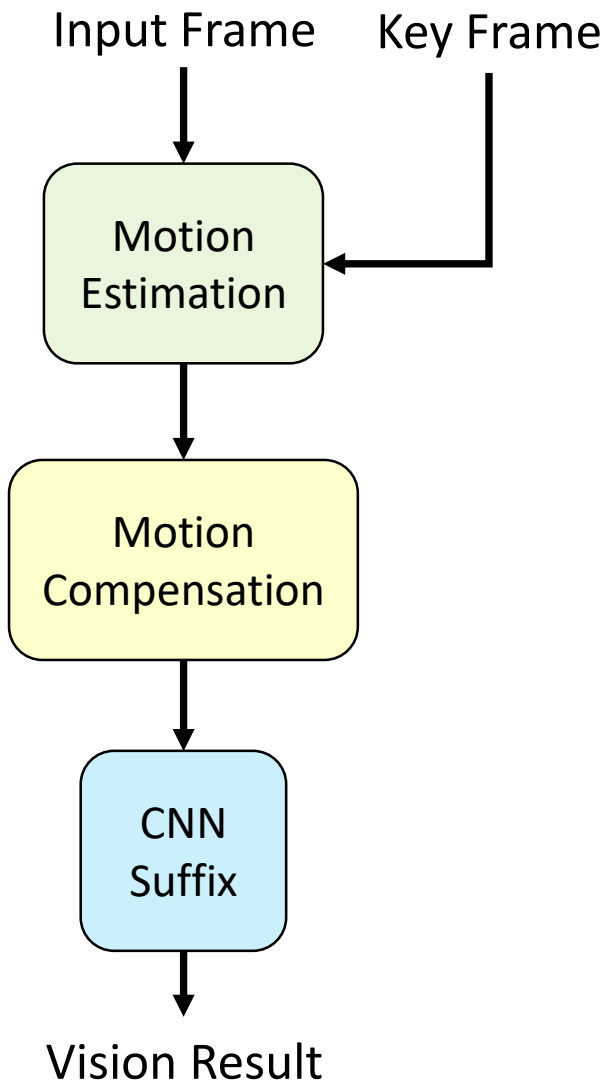
EVA^2 Energy Savings



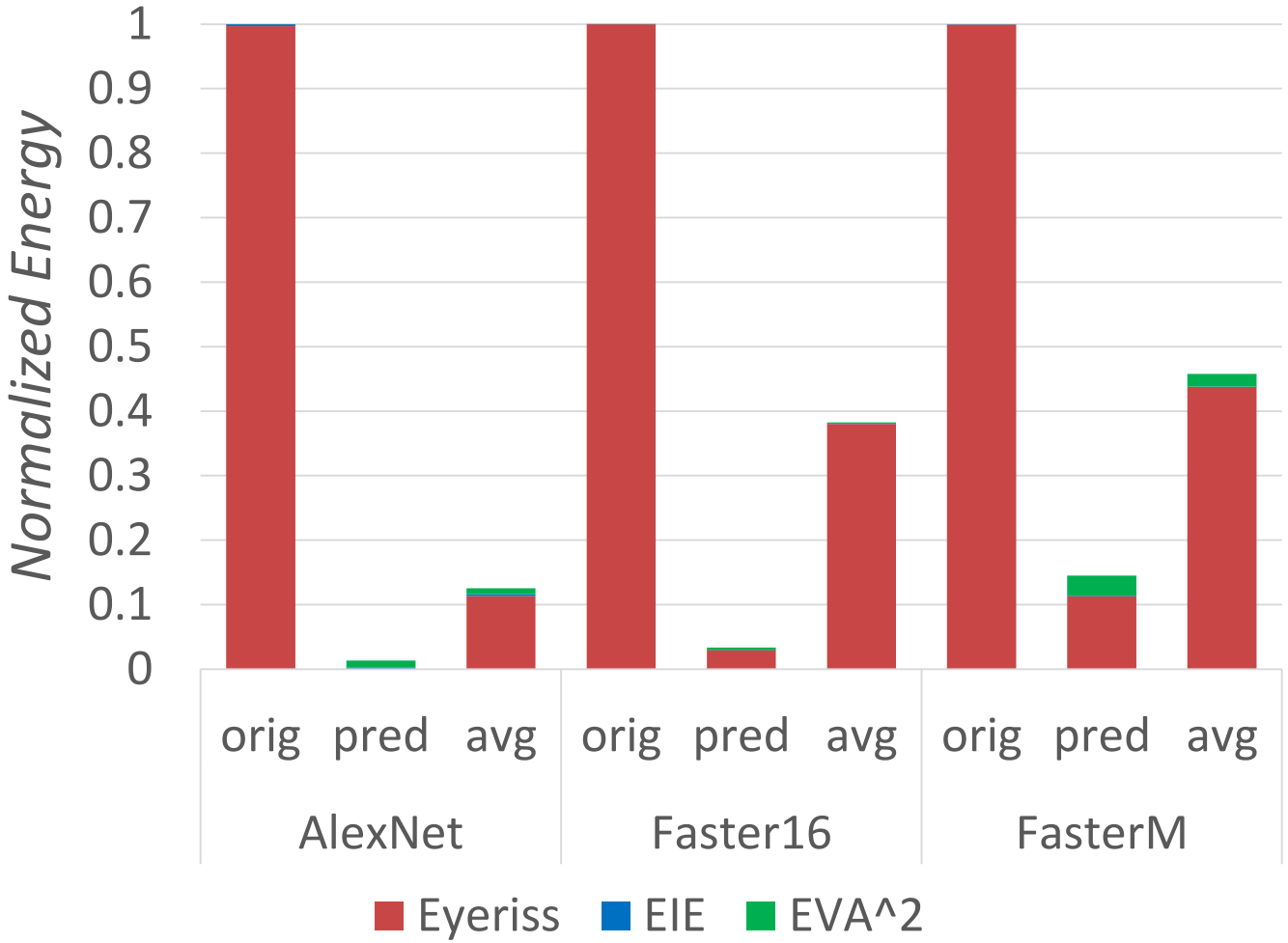


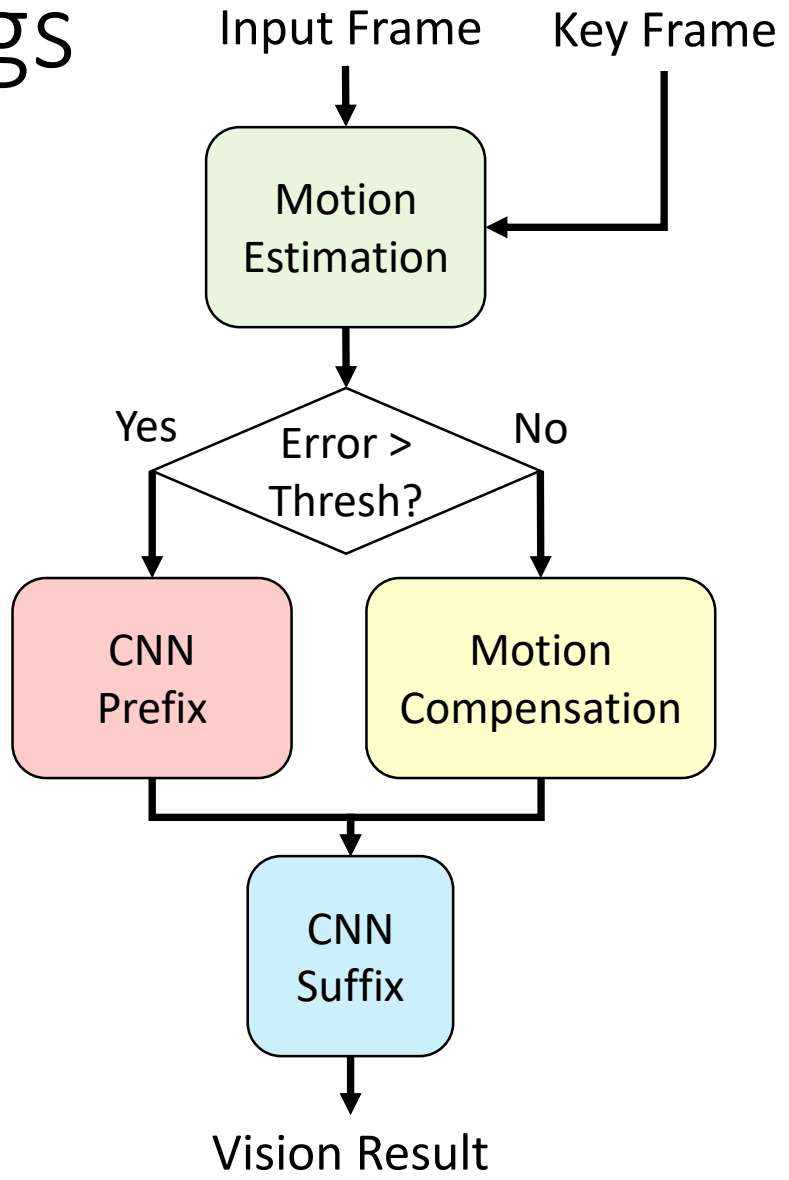
EVA^2 Energy Savings





EVA^2 Energy Savings





High Level EVA^2 Results

Network	Vision Task	Keyframe %	Accuracy Degredation	Average Latency Savings	Average Energy Savings
AlexNet	Classification	11%	0.8% top-1	86.9%	87.5%
Faster R-CNN VGG16	Detection	36%	0.7% mAP	61.7%	61.9%
Faster R-CNN VGGM	Detection	37%	0.6% mAP	54.1%	54.7%

- EVA^2 enables **54-87% savings** while incurring **<1% accuracy degradation**
- Adaptive key frame choice metric can be adjusted

Talk Overview

Background

Algorithm

Hardware

Evaluation

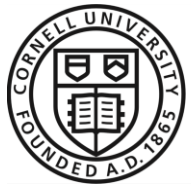
Conclusion

Conclusion

- Temporal redundancy is an entirely new dimension for optimization
- AMC & EVA^2 **improve efficiency** and are **highly general**
- Applicable to many different…
  - CNN applications (classification, detection, segmentation, etc)
  - Hardware architectures (CPU, GPU, ASIC, etc)
  - Motion estimation/compensation algorithms

EVA^2: Exploiting Temporal Redundancy In Live Computer Vision

Mark Buckler, Philip Bedoukian, Suren Jayasuriya, Adrian Sampson



Cornell University

ASU Arizona State University

International Symposium on Computer Architecture (ISCA)

Tuesday June 5, 2018

Backup Slides

Why not use vectors from video codec/ISP?

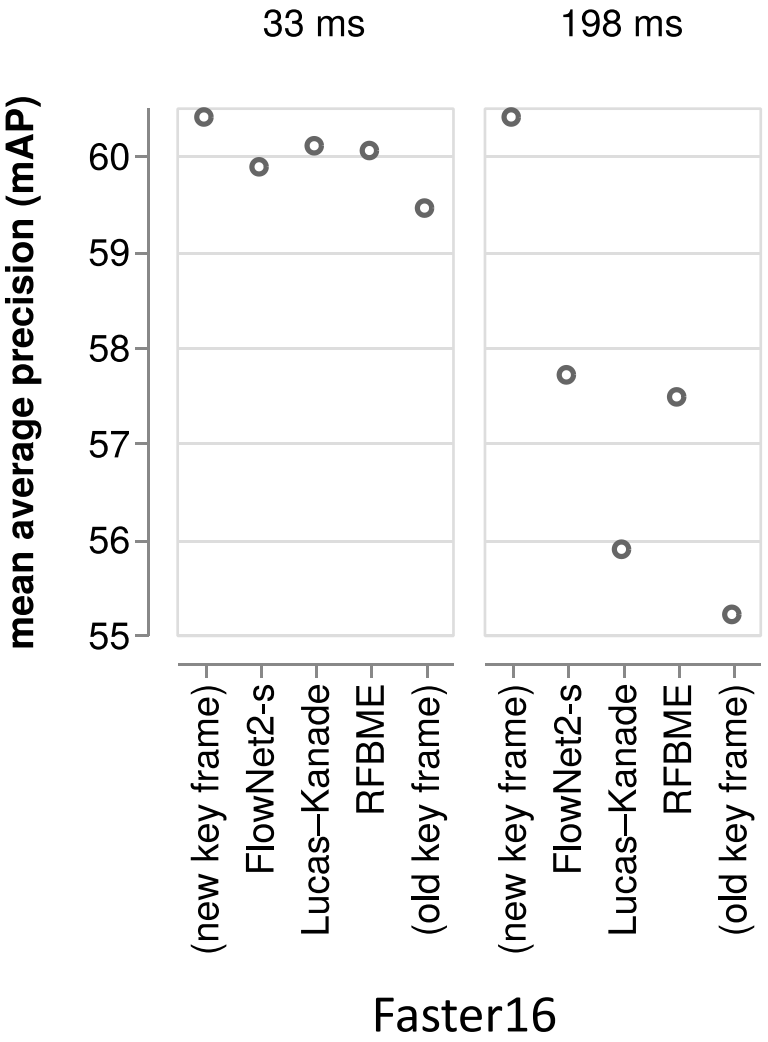
- We've demonstrated that the ISP can be skipped (Bucker et al. 2017)
  - No need to compress video which is instantly thrown away
  - Can save energy by power gating the ISP
  - Opportunity to set own key frame schedule

- However, great idea for pre-stored video!

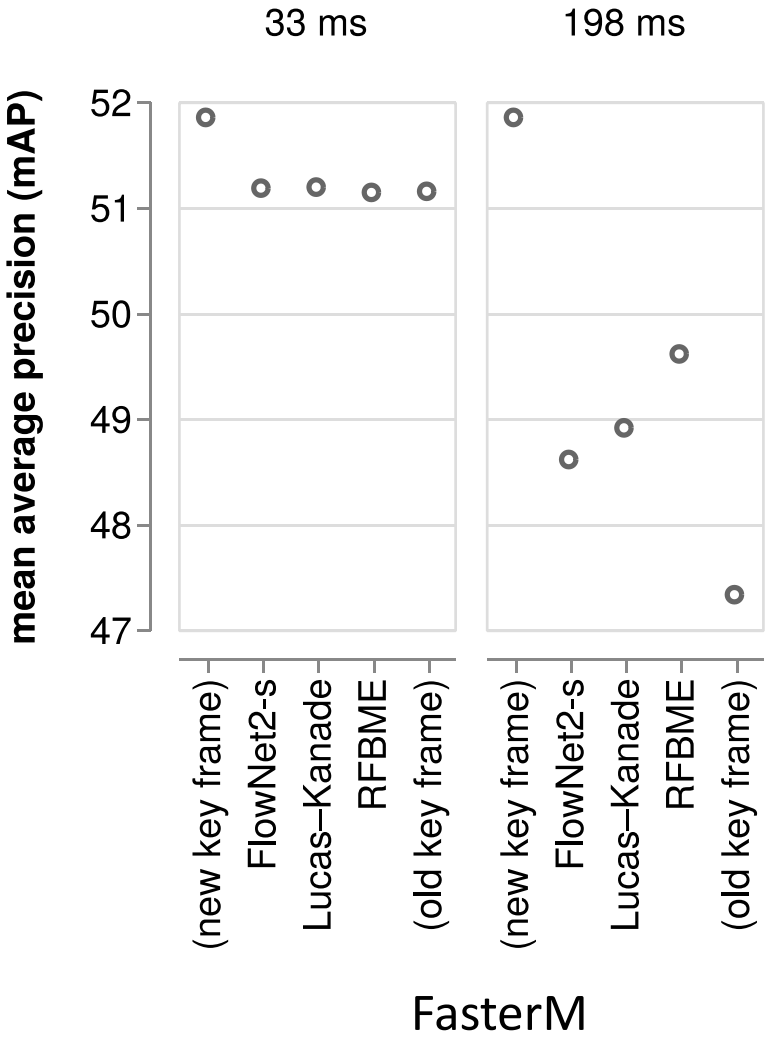
Why Not Simply Subsample?

- If lower frame rate needed, simply apply AMC at that frame rate
  - Warping
  - Adaptive key frame choice

Different Motion Estimation Methods



Faster16



FasterM

Difference from Deep Feature Flow?

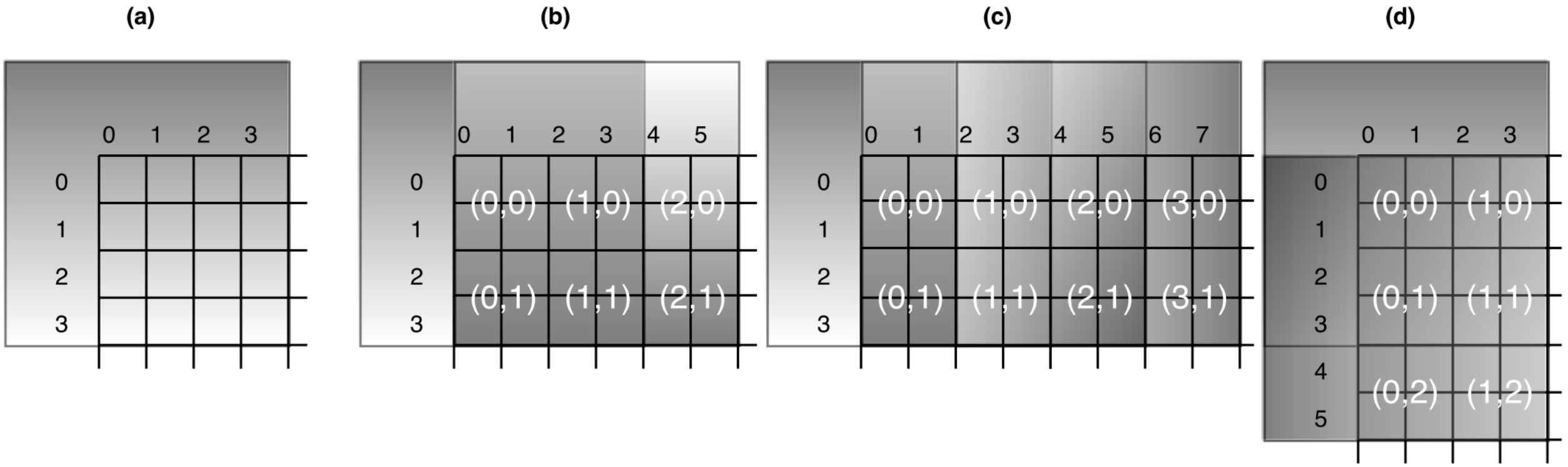
- Deep Feature Flow does also exploit temporal redundancy, but…

	AMC and EVA^2	Deep Feature Flow
Adaptive key frame rate?	Yes	No
On chip activation cache?	Yes	No
Learned motion estimation?	No	Yes
Motion estimation granularity	Per receptive field	Per pixel (excess granularity)
Motion compensation	Sparse (four-way zero skip)	Dense
Activation storage	Sparse (run length)	Dense

Difference from Euphrates?

- Euphrates has a strong focus on SoC integration
- Motion estimation from ISP
  - May want to skip the ISP to save energy & create more optimal key schedule
- Motion compensation on bounding boxes
  - Skips entire network, but is only applicable to object detection

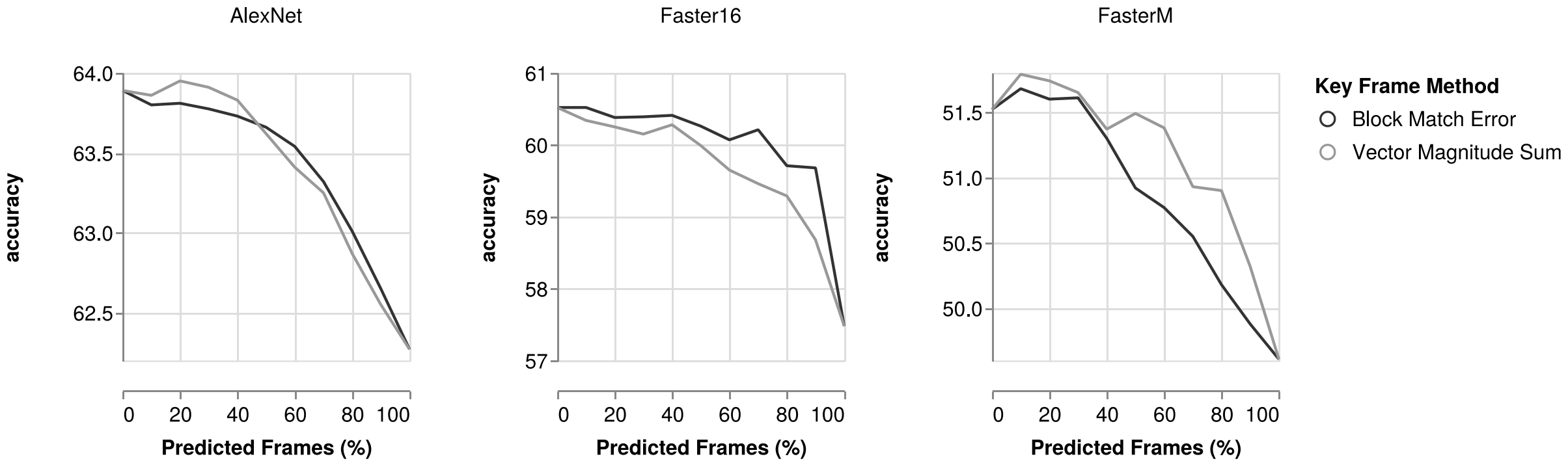
Re-use Tiles in RFBME



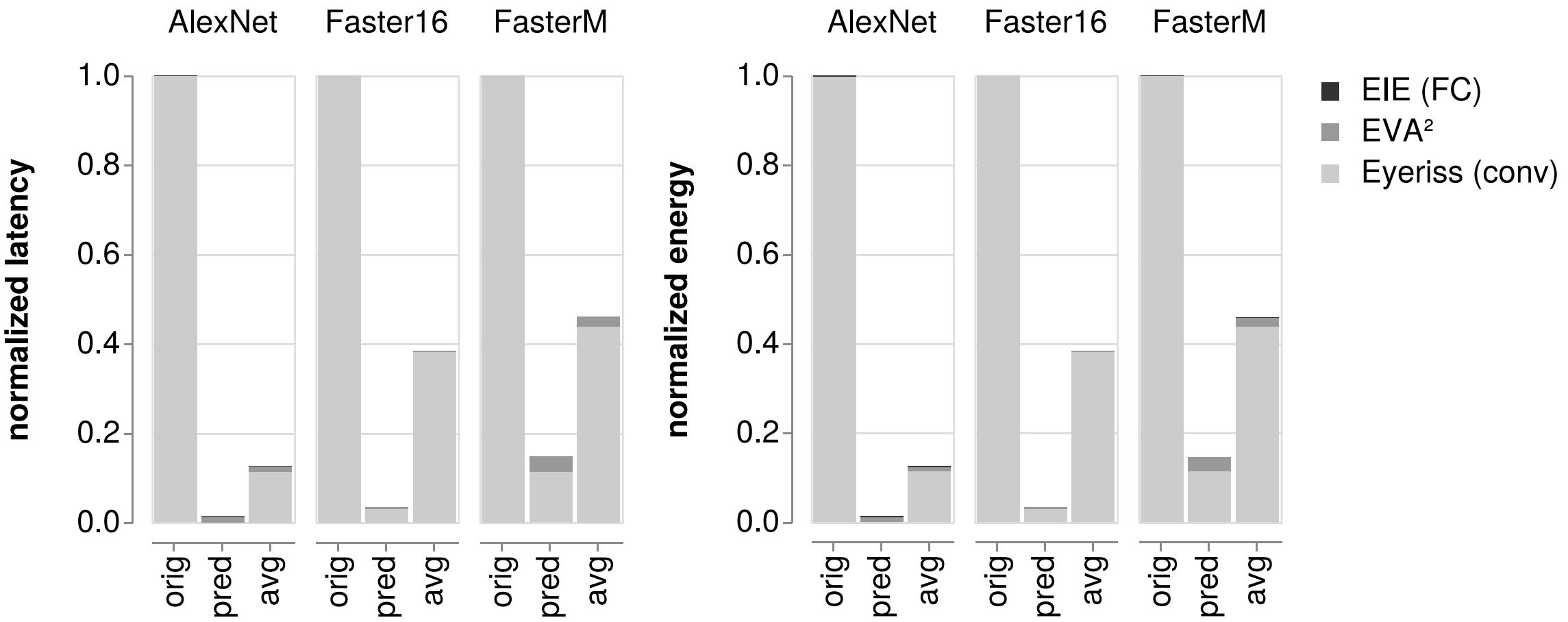
Changing Error Threshold

Network	Config	Acc.	Keys	Time (ms)	Energy (mJ)
AlexNet	orig	65.1	100%	115.4	32.2
	hi	65.1	22%	26.7	7.4
	med	64.3	11%	14.5	4.0
	lo	63.8	4%	5.9	1.6
Faster16	orig	60.1	100%	4370.1	1035.5
	hi	60.0	60%	2664.8	631.3
	med	59.4	36%	1673.6	396.4
	lo	58.9	29%	1352.7	320.3
FasterM	orig	51.9	100%	492.3	116.7
	hi	51.6	61%	327.2	77.4
	med	51.3	37%	226.4	53.4
	lo	50.4	29%	194.7	45.9

Different Adaptive Key Frame Metrics



Normalized Latency & Energy



How about Re-Training?

Network	Target Layer	Accuracy
FasterM	No Retraining	51.02
	Early Target	45.35
	Late Target	47.82
Faster16	No Retraining	60.4
	Early Target	61.30
	Late Target	60.52

Where to cut the network?

Network	Interval	Early Target	Late Target
AlexNet	orig	63.52	63.52
	4891 ms	49.95	53.64
Faster16	orig	60.4	60.4
	33 ms	60.29	60.05
	198 ms	55.44	57.48
FasterM	orig	51.85	51.85
	33 ms	50.90	51.14
	198 ms	48.77	49.61

#MakeRyanGoslingTheNewLenna

- Lenna dates back to 1973
- We need a new test image for image processing!