Implementation and evaluation of attention layer on gaze tracking using neural network

Yucheng XIAO

Supervisor Professor: Renaud Lachaize

Supervisor in Lab: Didier Schwab

Hosted by GETALP Team

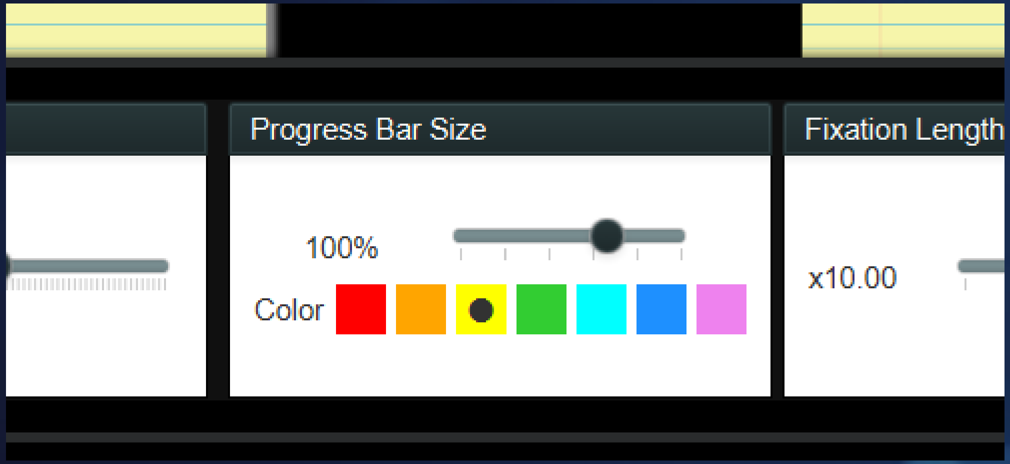
Outline

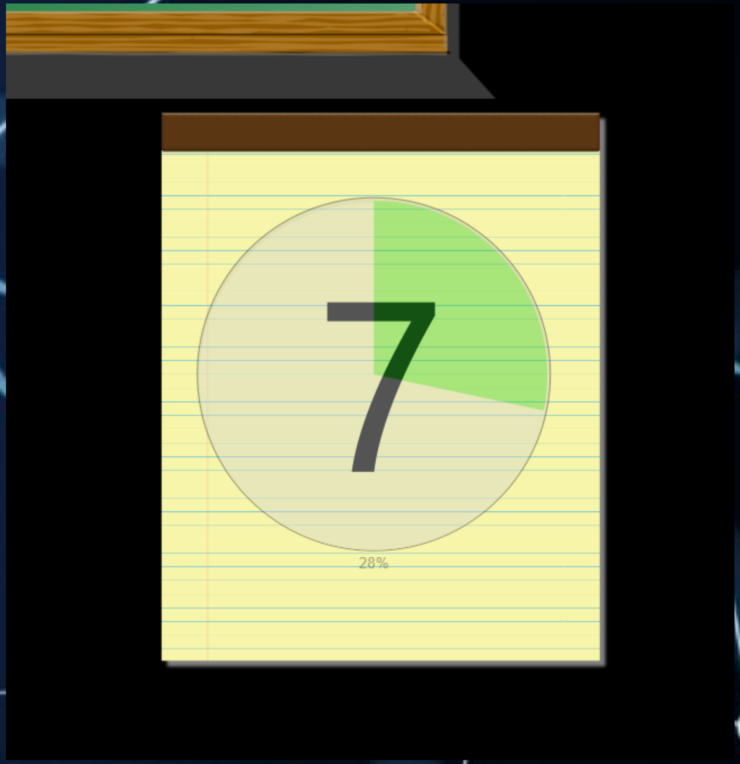
1. Issue correction on Gazeplay games
2. Performance evaluation and comparison on gaze tracking of webcam and eye-tracker
3. Implementation and evaluation of attention layer on gaze tracking using neural network

Issue correction on Gazeplay games

Issue #1572 :

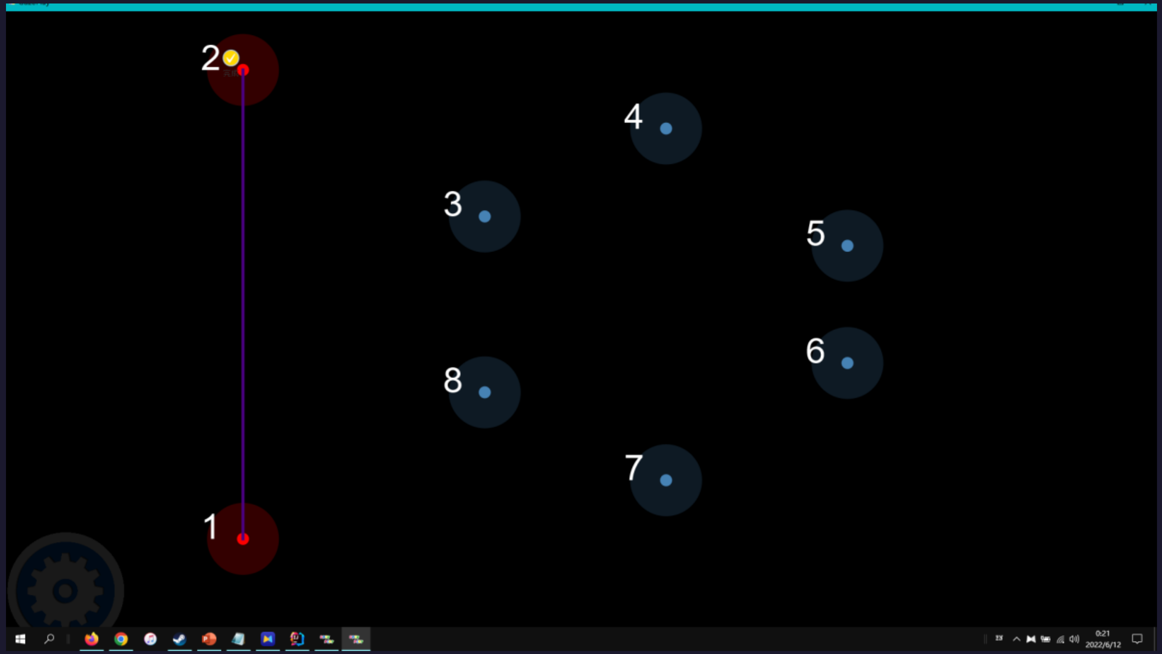
Add the option to modifty progress bar size & color

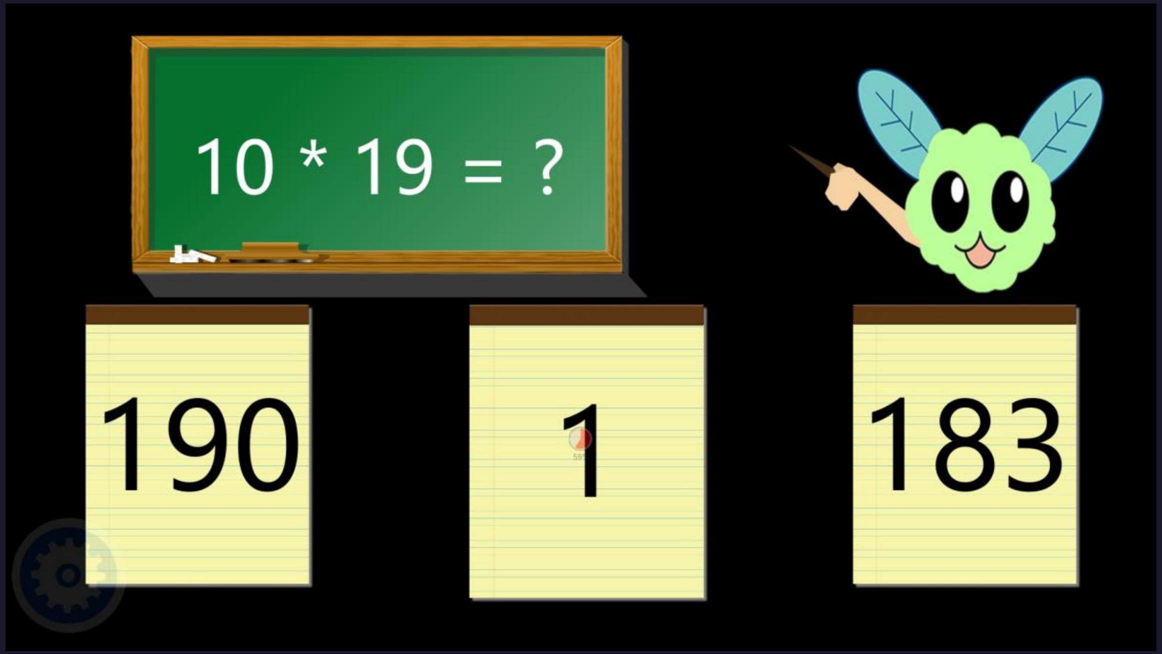




Performance evaluation and comparison on webcam and eye-tracker

The game Gazeplay





https://gazeplay.github.io/GazePlay/

Some comparisions…

Laptop camera

- Free (Basiclly)
- Integrated in laptop
- Drivers already installed

Tobii Eye Tracker 5

- 259.00€
- Fixtation required
- Driver installation required

Can we use laptop camera instead ?

Defining Performance metrics

- Accuracy
- Precision
- Gaze Time

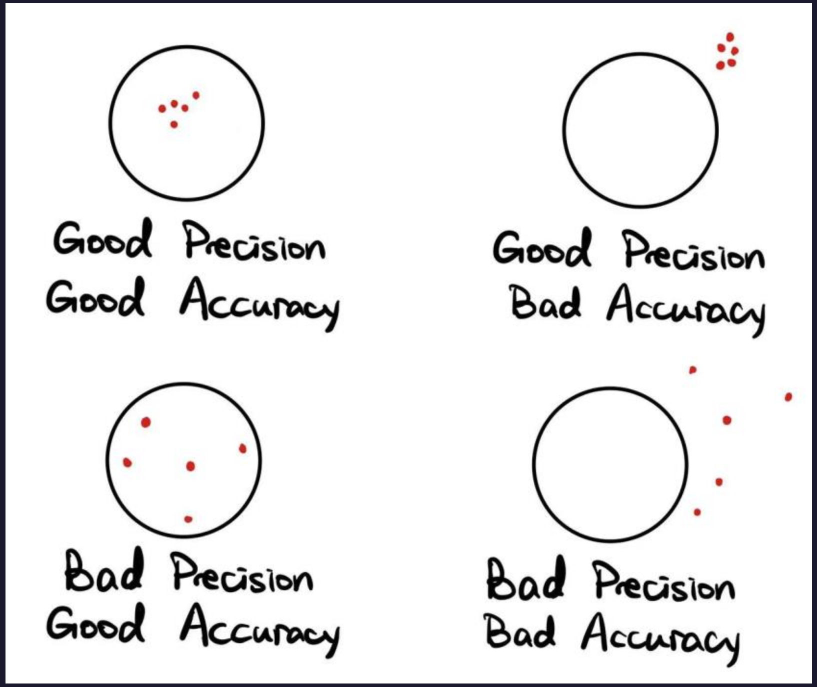
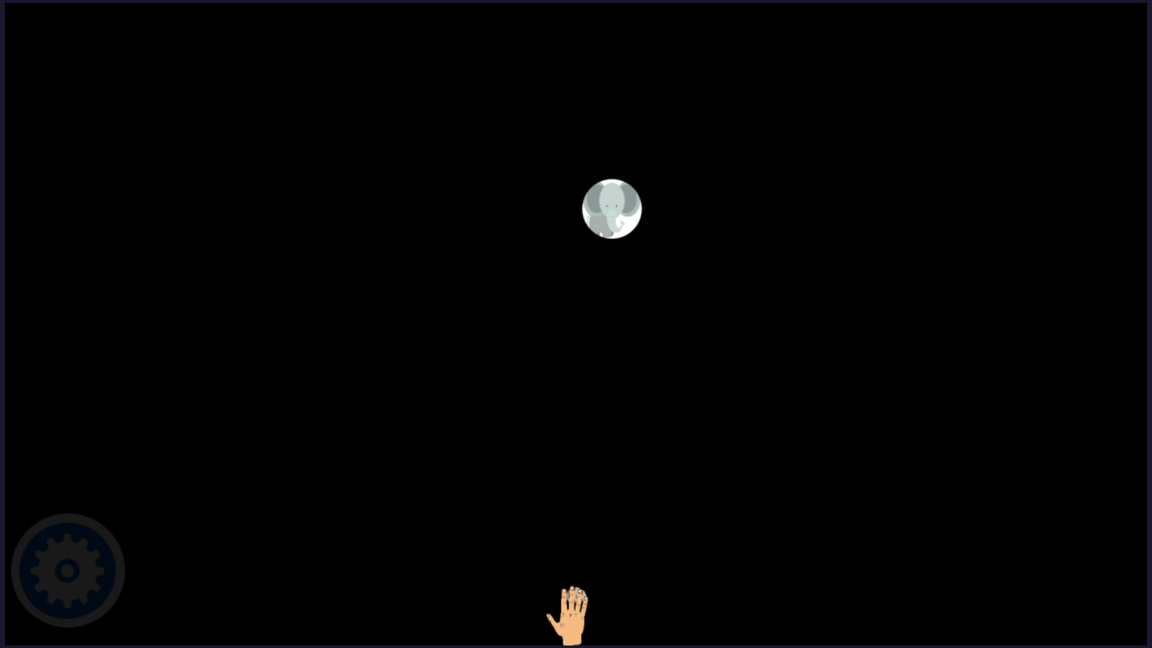
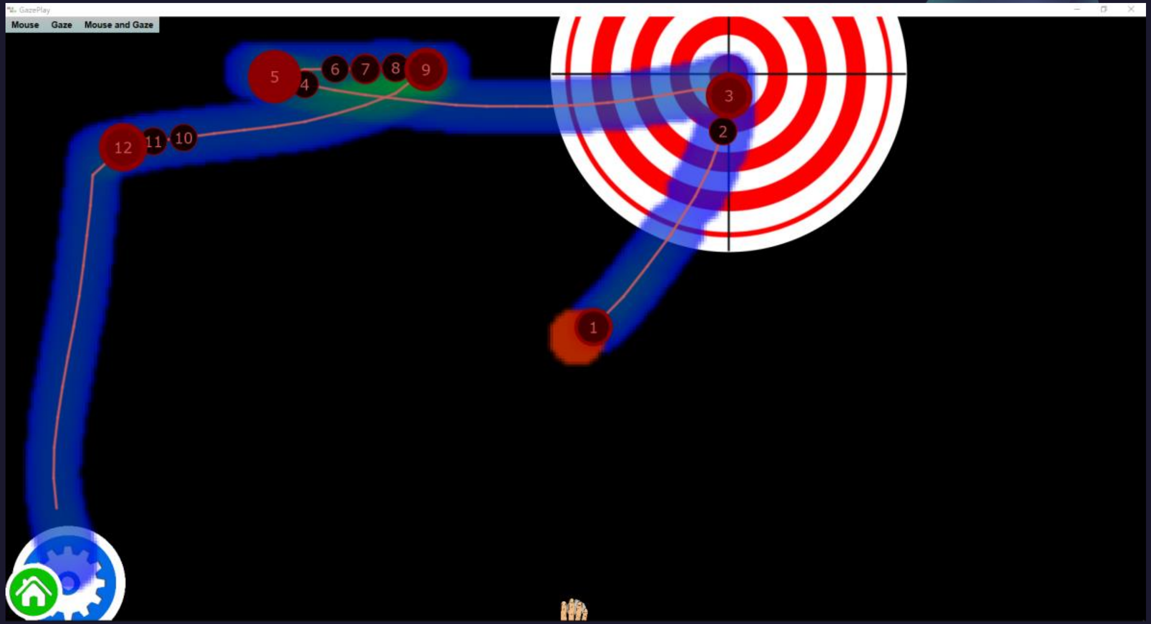


Figure inspired by [Tobii, 2011]

Test Environments

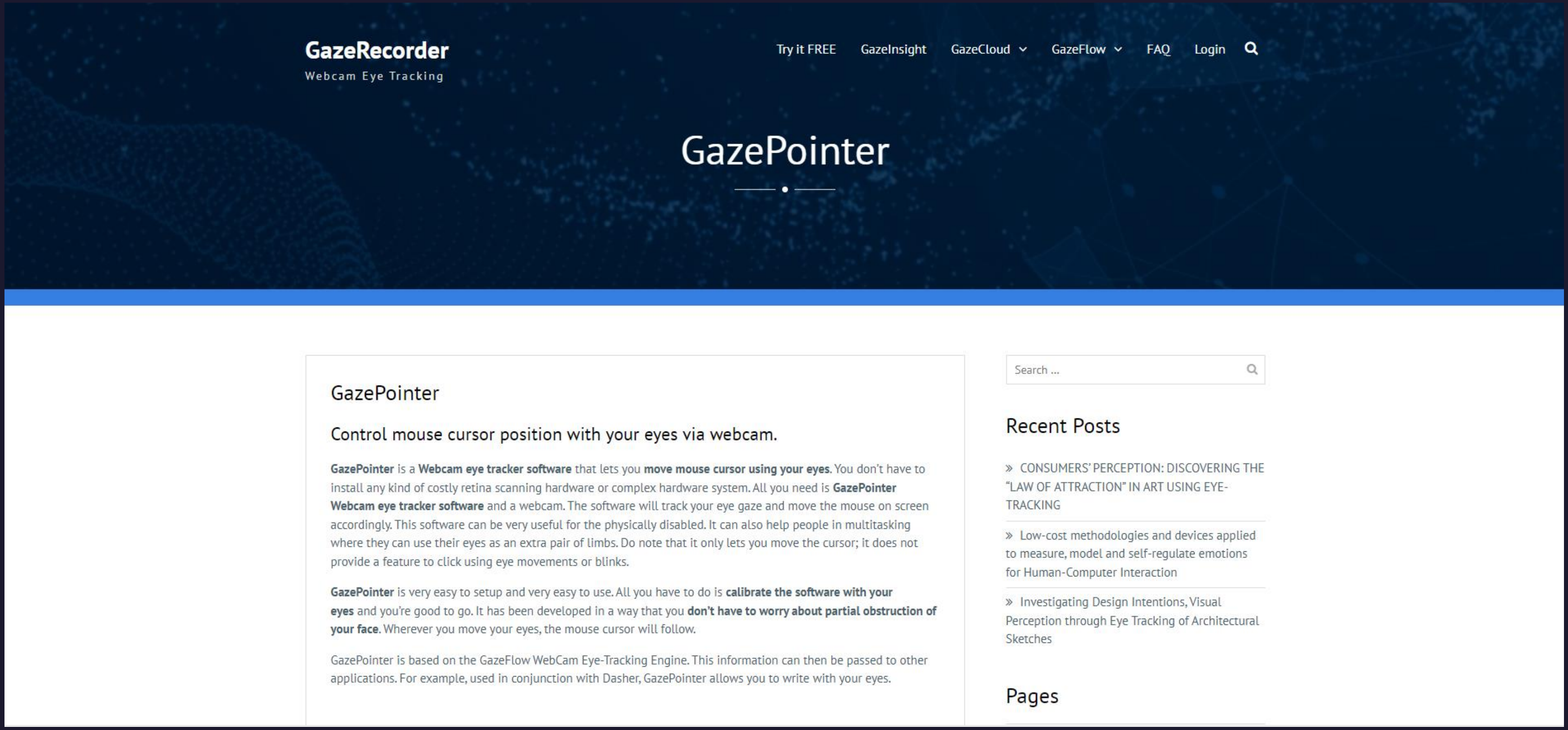
The Creampie Game – With modifications





Test Environments on laptop camera

Application GazePointer



https://gazerecorder.com/gazepointer/

Test Results

	Precision pixels	Accuracy pixels	Gaze Time ms
Eye-Tracker	29	36	430
Webcam	58	112	989

Table 1 : Evaluation results

Implementation and evaluation of attention layer on gaze tracking using neural network

What is Attention ?

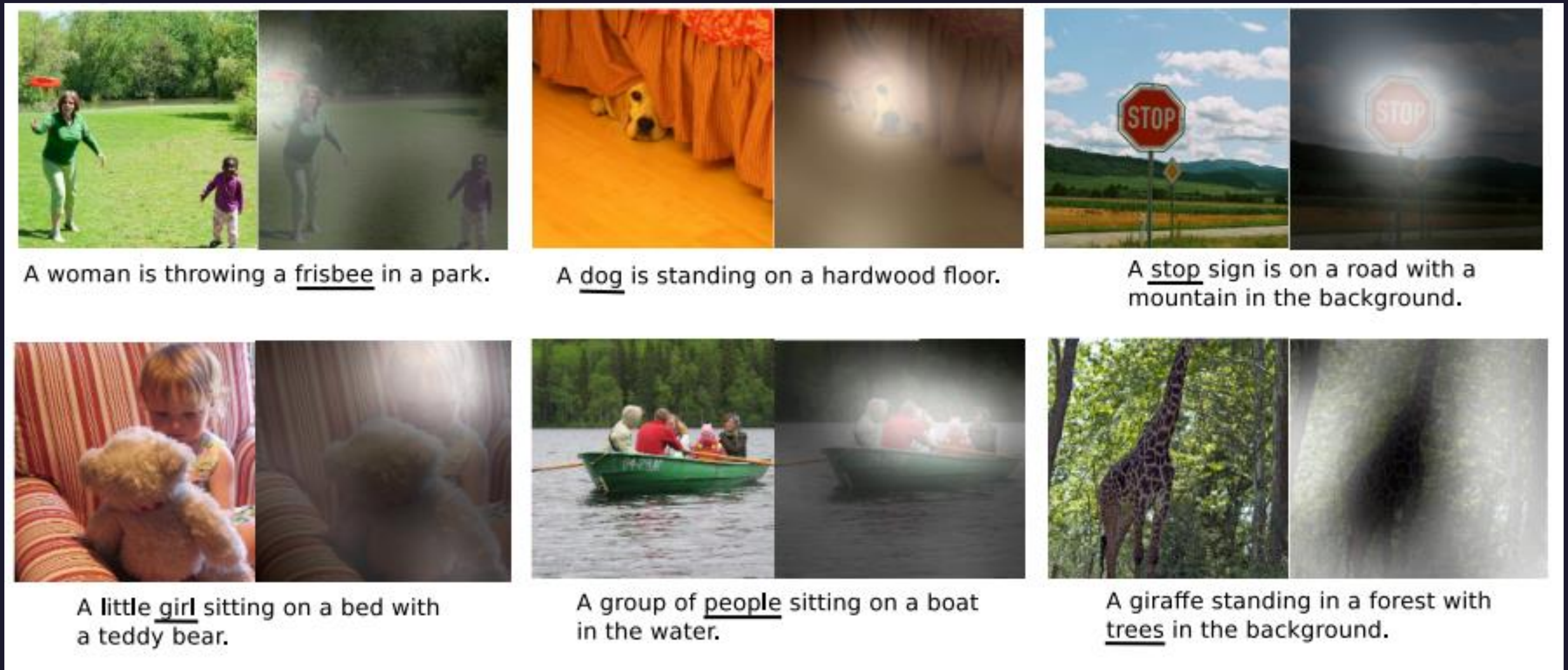


Image from [Xu et al., 2015]

Widely Used Attention

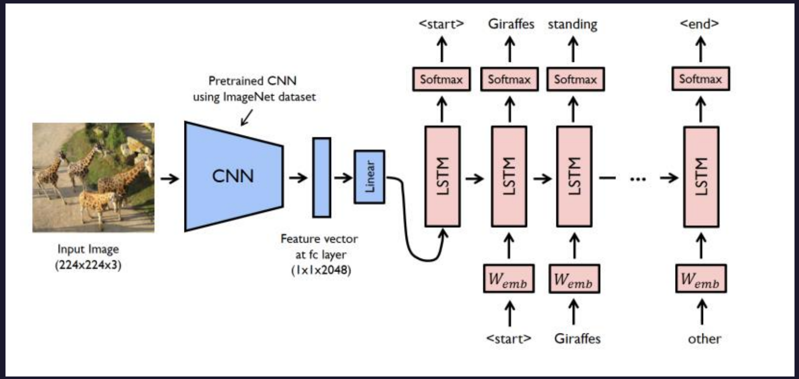
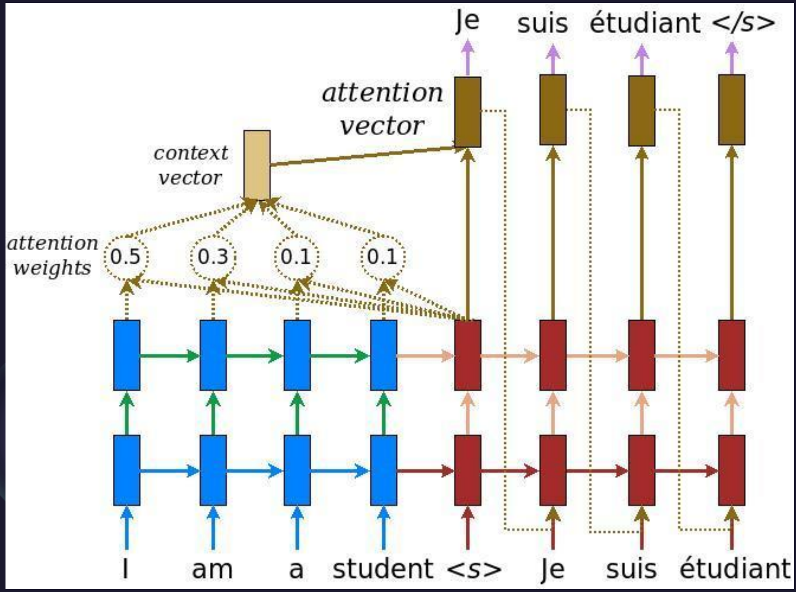


Image captioning



Language translation

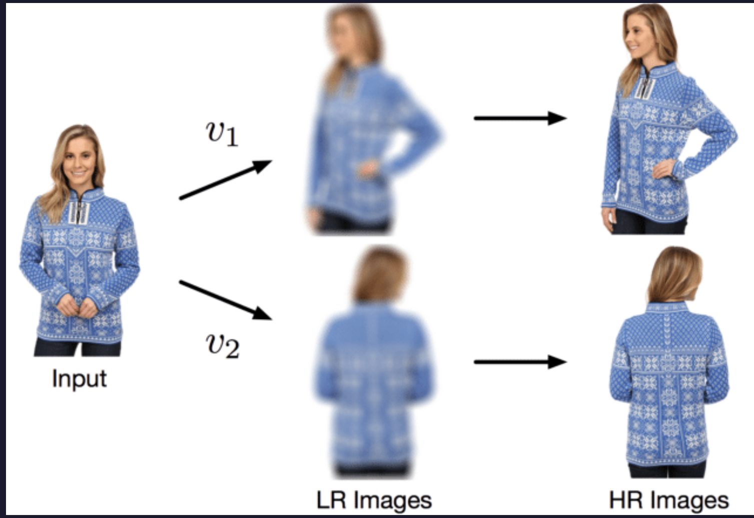


Image Generation

Database : MPIIGaze

- Pictures from laptop cameras

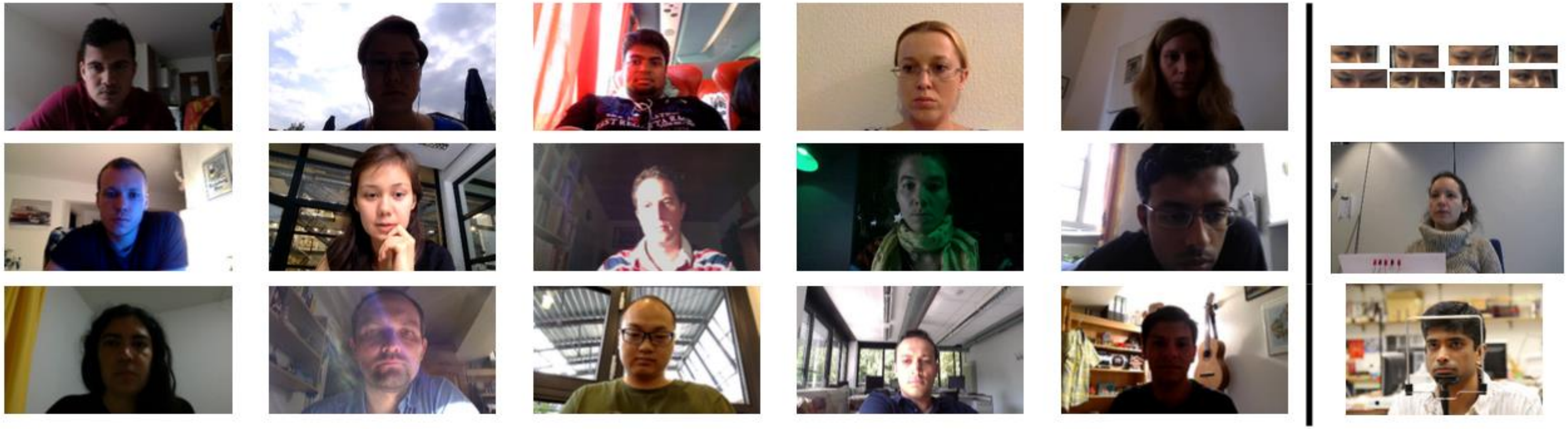
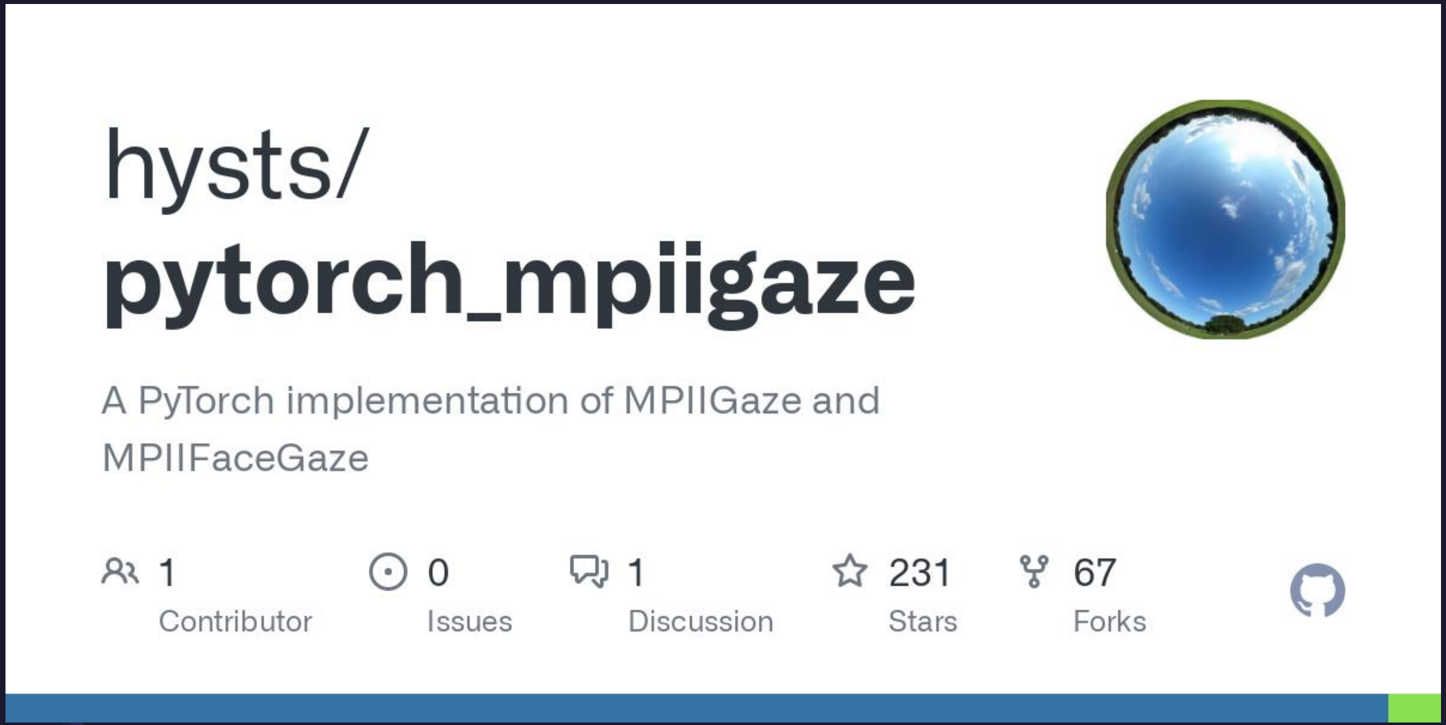


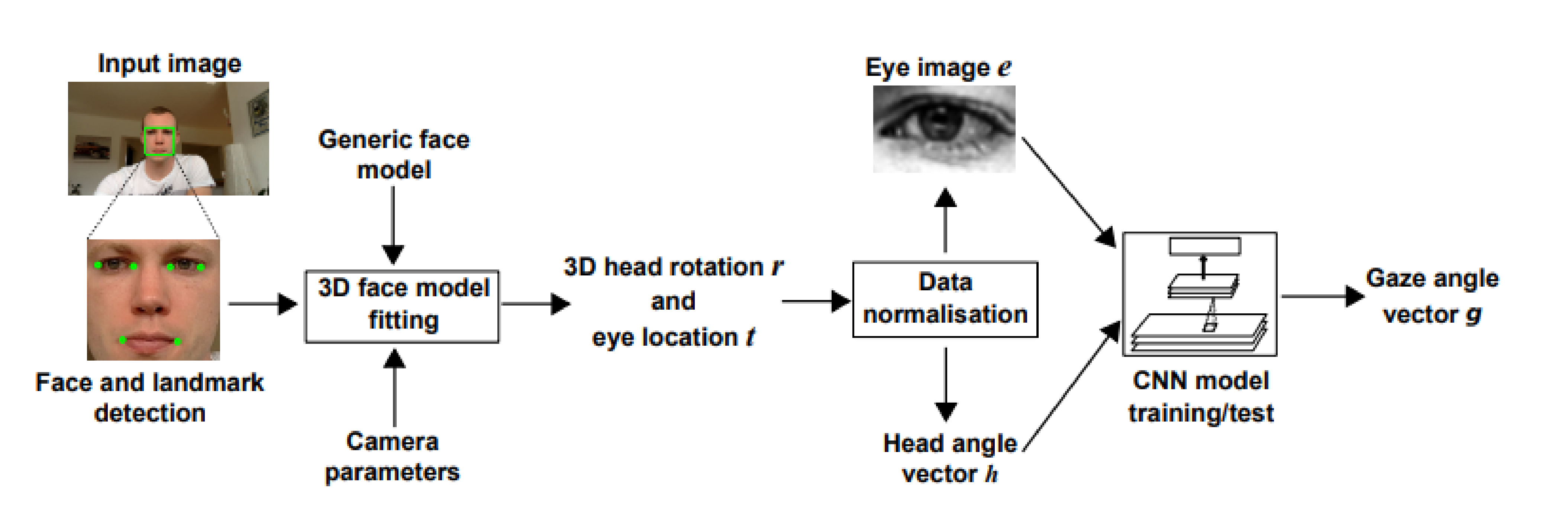
Figure 2: Sample images from our MPIIGaze dataset showing the considerable variability in terms of place and time of recording, directional light and shadows. For comparison, the last column shows sample images from other current publicly available datasets (cf. Table 1): UT Multiview [39] (top), Eyediap [8] (middle), Smith et al. [37] (bottom).

Figure from [Zhang et al., 2015]

Pytorch-MPIIGaze

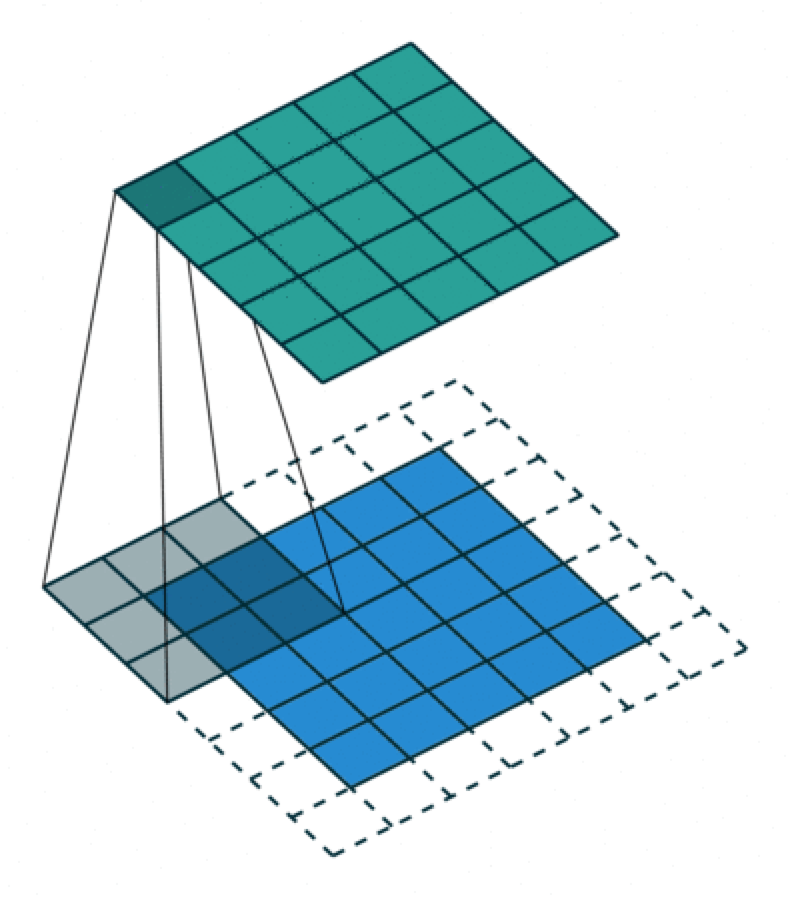


Pytorch-MPIIGaze



Gaze angle calculation process of pytorch-mpiigaze
Figure from [Zhang et al., 2015]

Convolutional Neural Networks

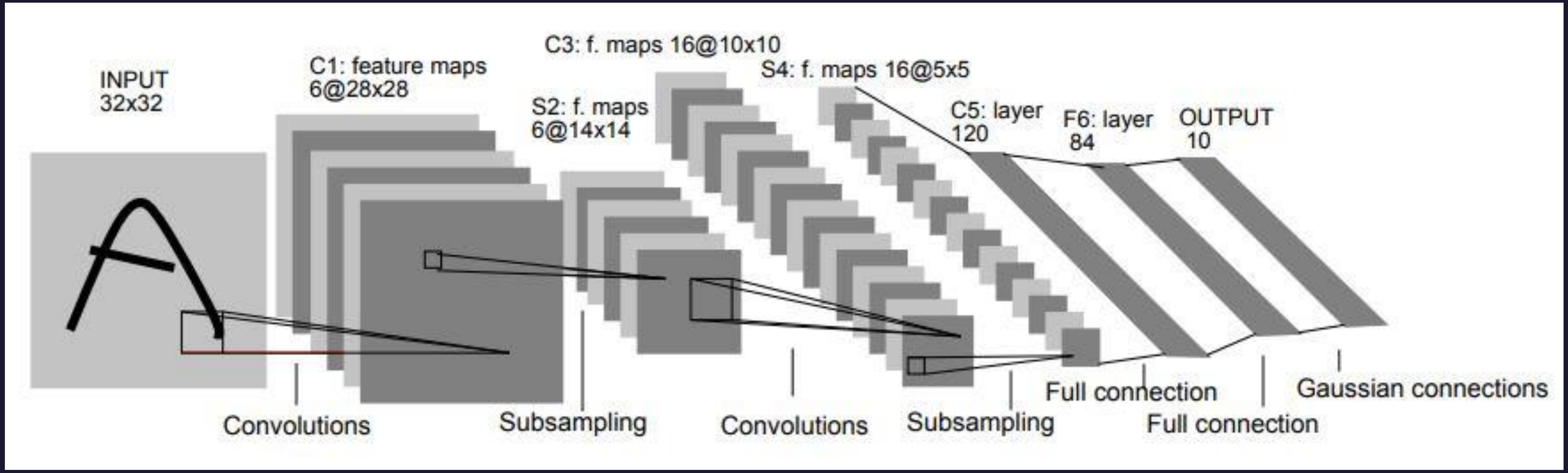


3x3 convolution, half padding

Animation from https://github.com/vdumoulin/conv_arithmetic/

LeNet

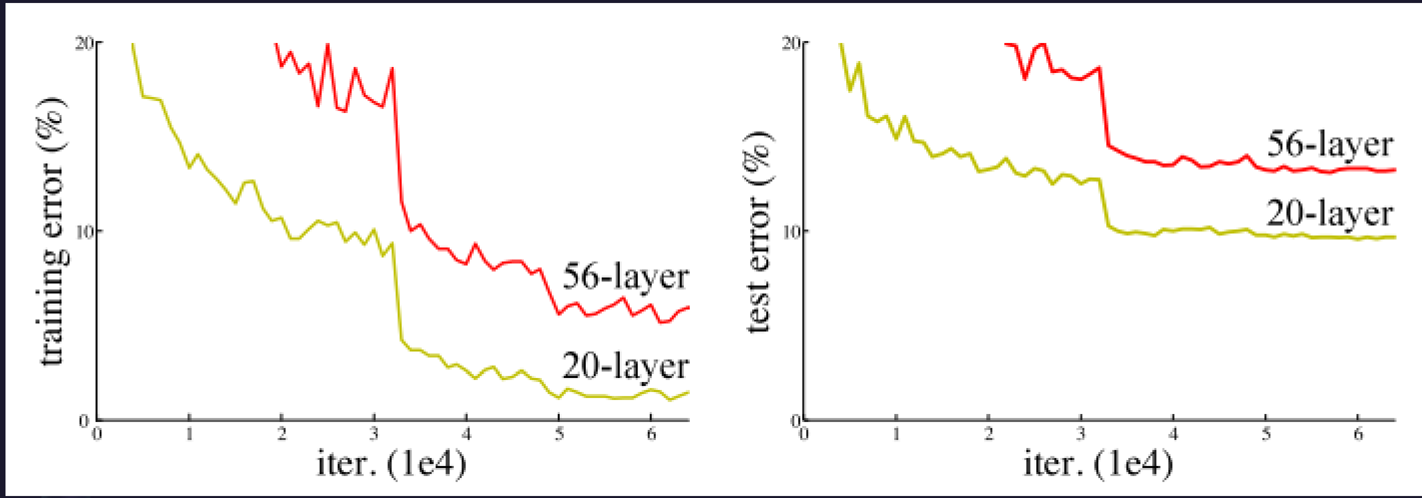
- Proposed by Yann Lecun and others in 1998



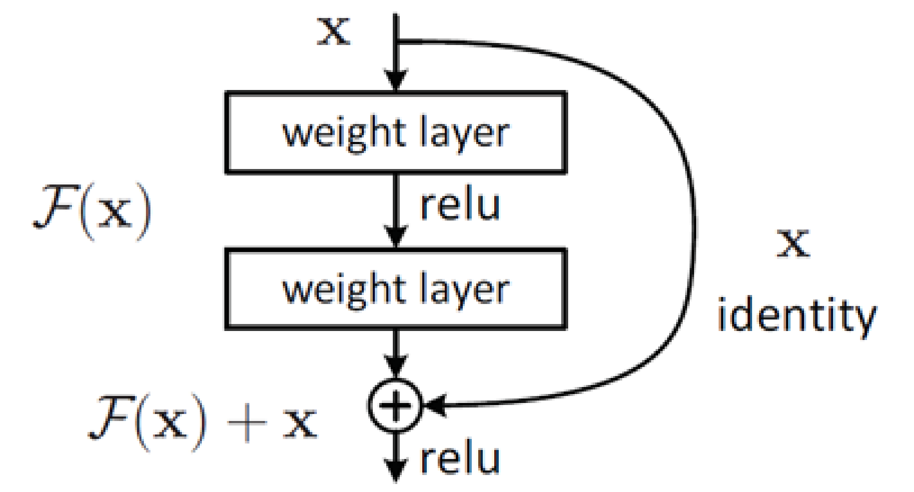
Structure of LeNet-5 from [Lecun et al., 1998]

ResNet

- Proposed by Kaiming He and others in 2015
- Winner of 2015 ILSVRC competition



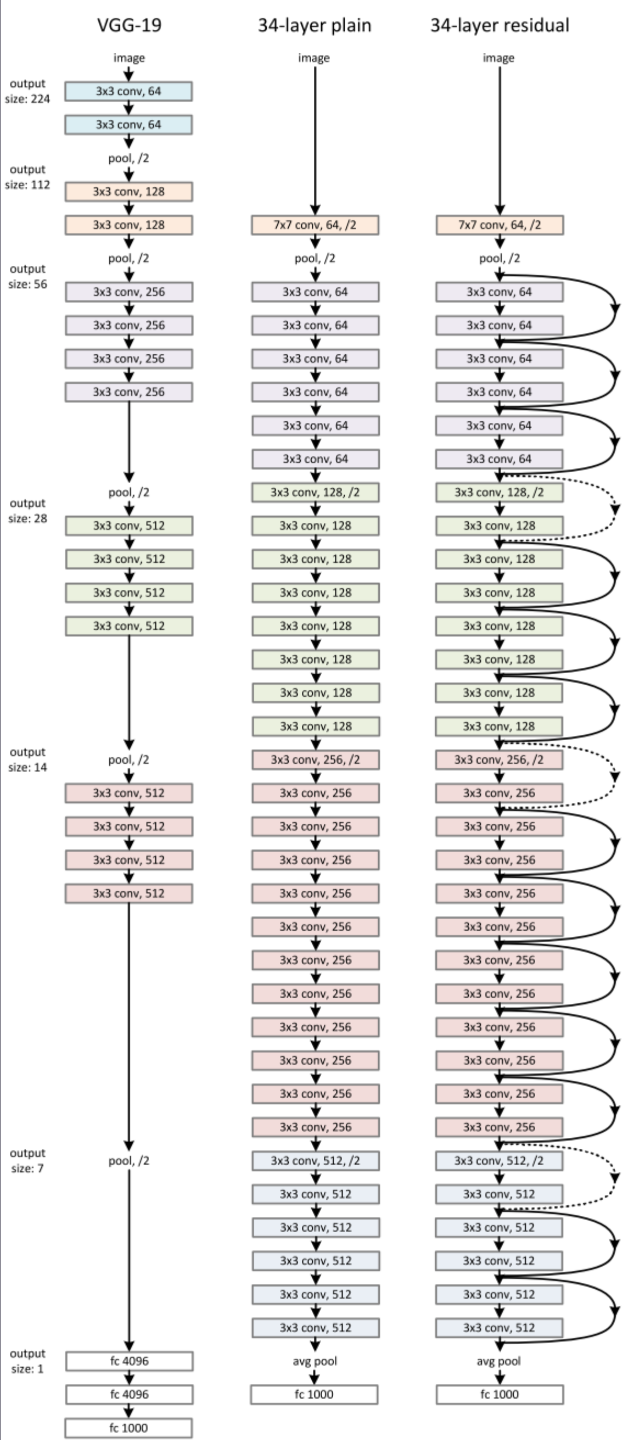
As the network goes deeper, its performance gets saturated or even starts degrading rapidly…



A residual block

Figures from [He et al., 2015]

ResNet



Left: the VGG-19 model (19.6 billion FLOPs) as a reference.

Middle: a plain network with 34 parameter layers (3.6 billion FLOPs).

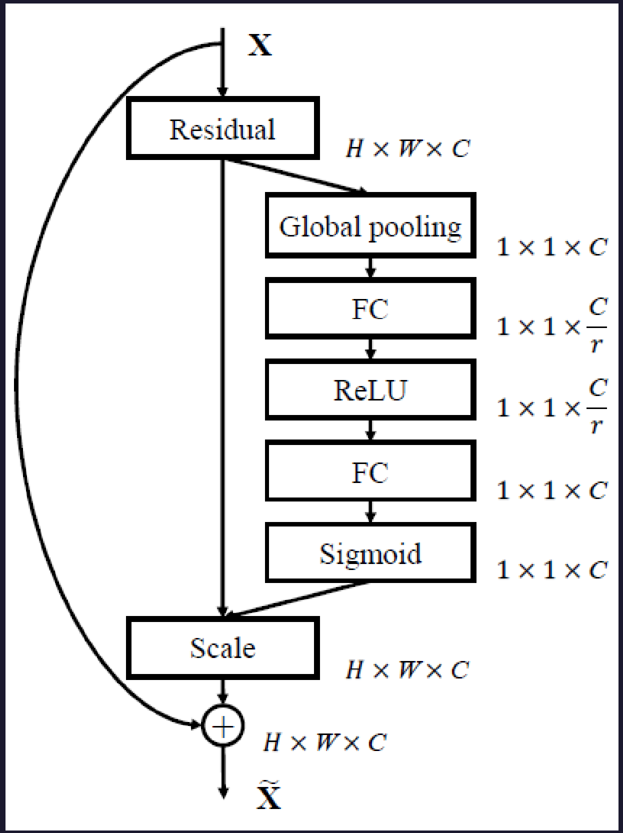
Right: a residual network with 34 parameter layers (3.6 billion FLOPs).

Figure from [He et al., 2015].

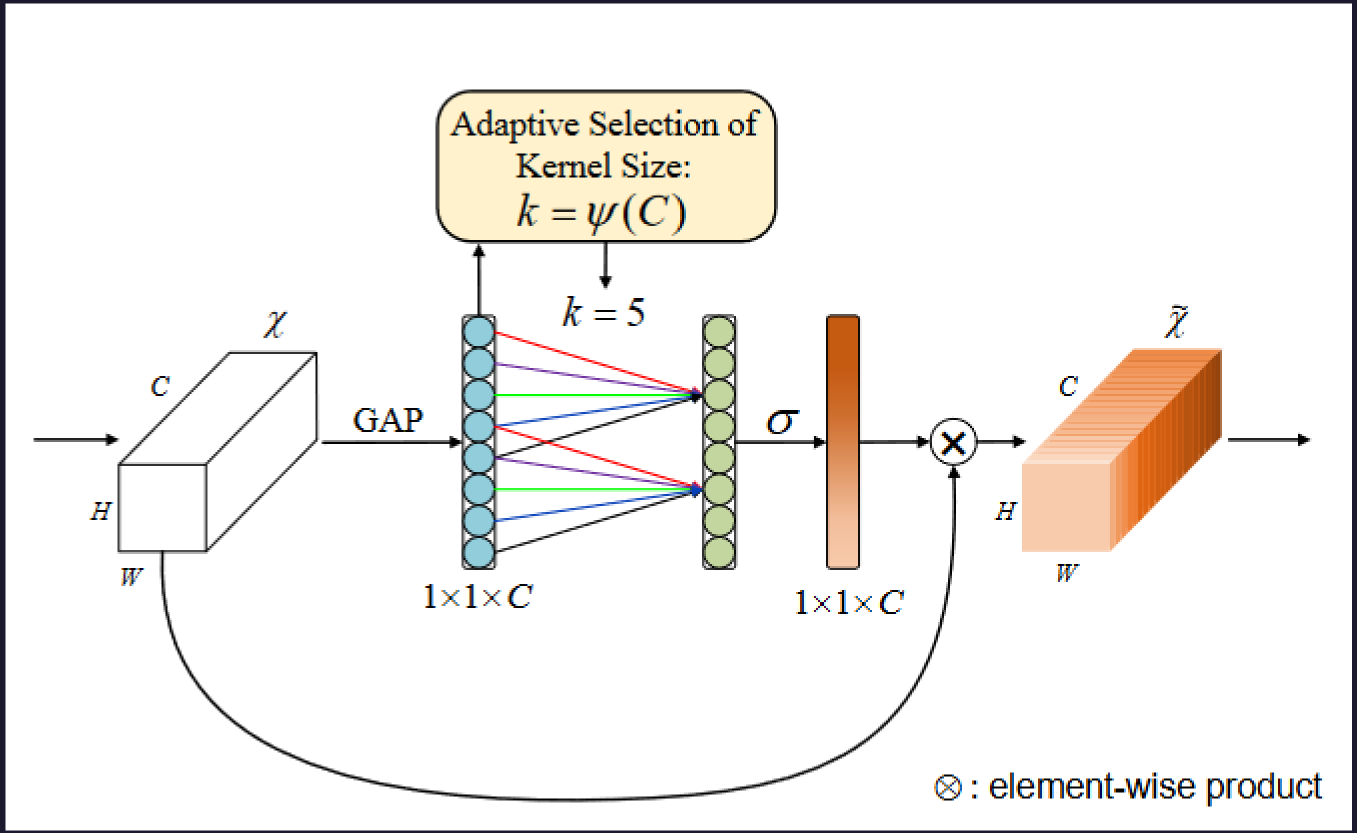
Attention layers

- ECA-Net, Efficient Channel Attention [Wang et al., 2019];
- Coordinate Attention [Hou et al., 2021];
- CBAM: Convolutional Block Attention Module [Woo et al., 2018];
- SK-Net: Selective Kernel Networks [Li et al., 2019].

ECA-Net

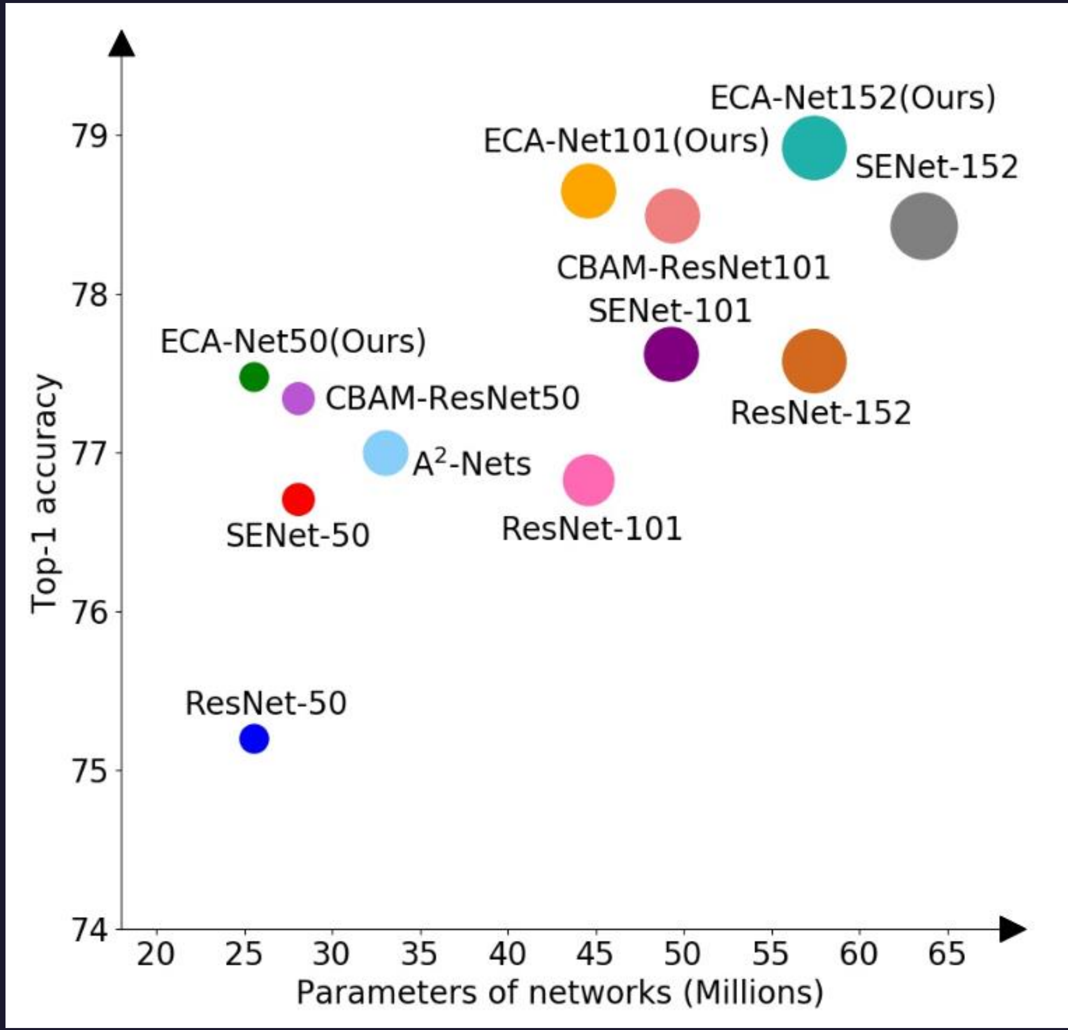


SE-Net, 2017



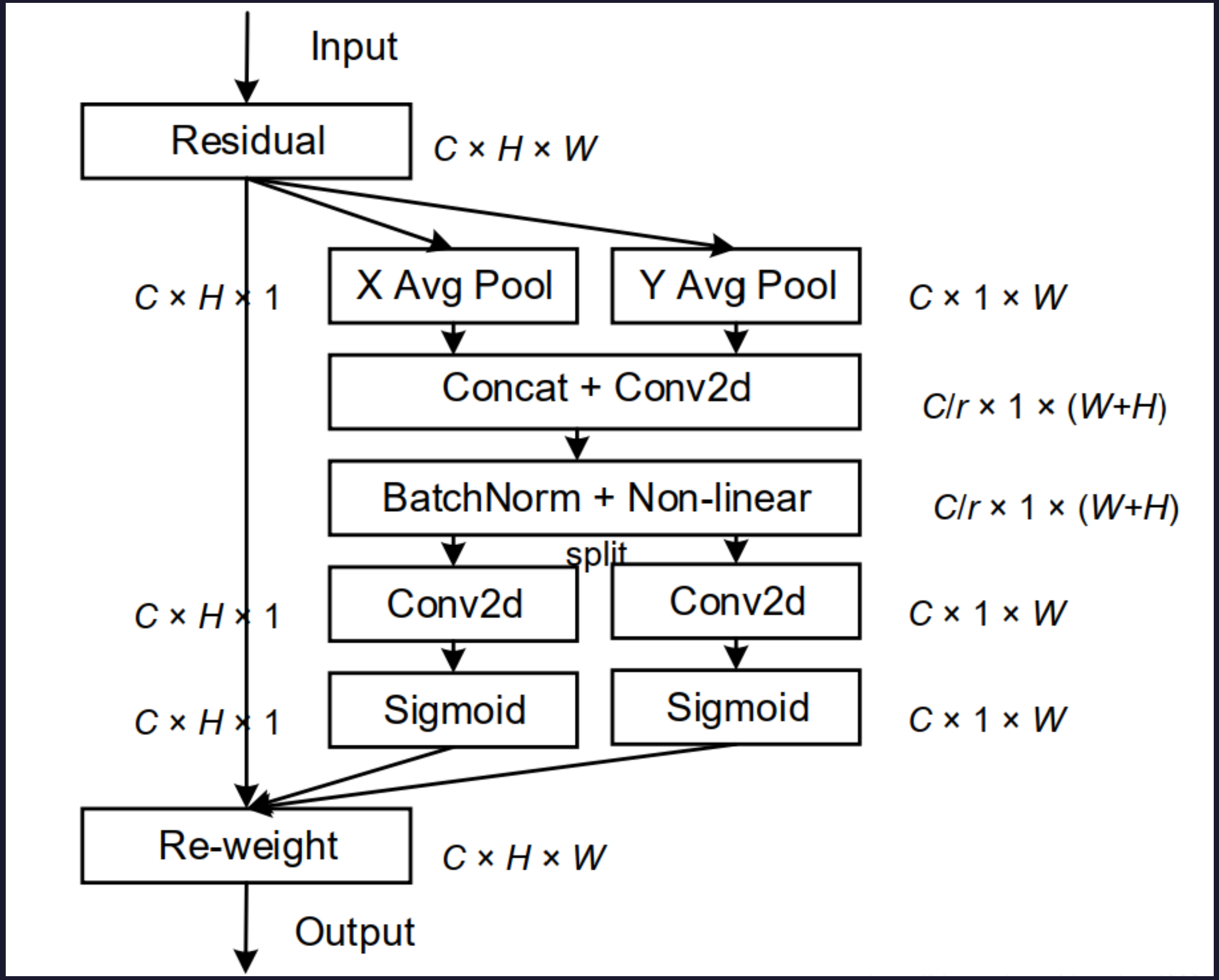
ECA-Net, 2019

ECA-Net

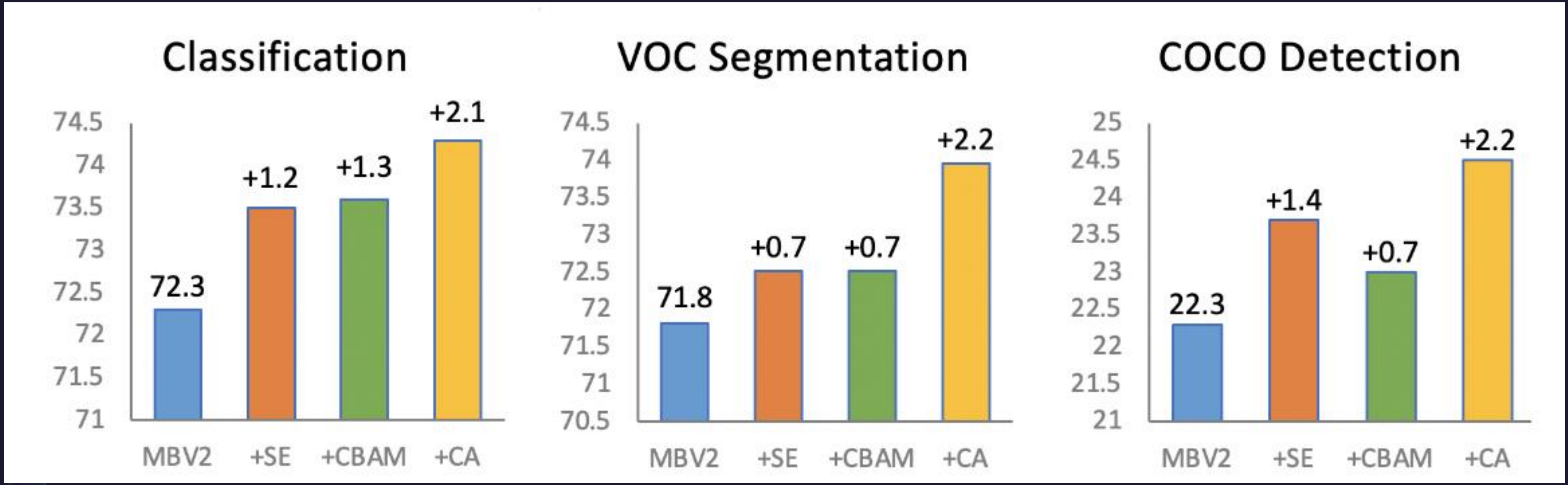


ECA-Net152(Ours)
ECA-Net101(Ours)
SENet-152
CBAM-ResNet101
SENet-101
ECA-Net50(Ours)
CBAM-ResNet50
ResNet-152
A2-Nets
SENet-50
ResNet-101
ResNet-50
Top-1 accuracy
79
78
77
76
75
74
20
25
30
35
40
45
50
55
60
65
Parameters of networks (Millions)

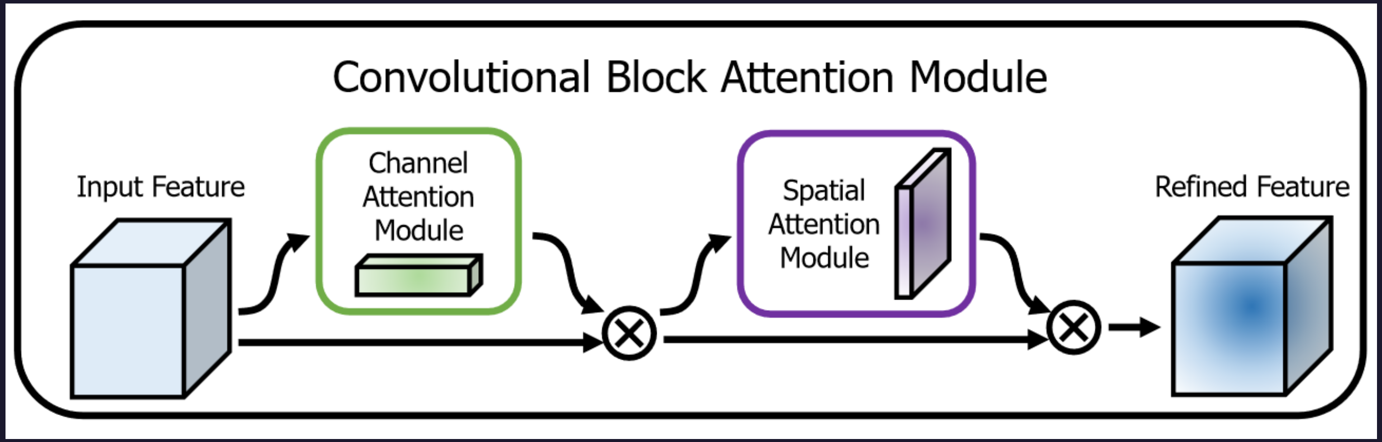
Coordinate Attention

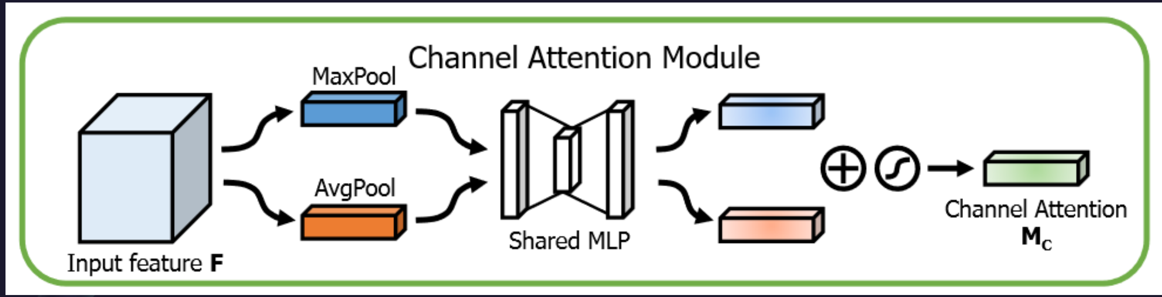


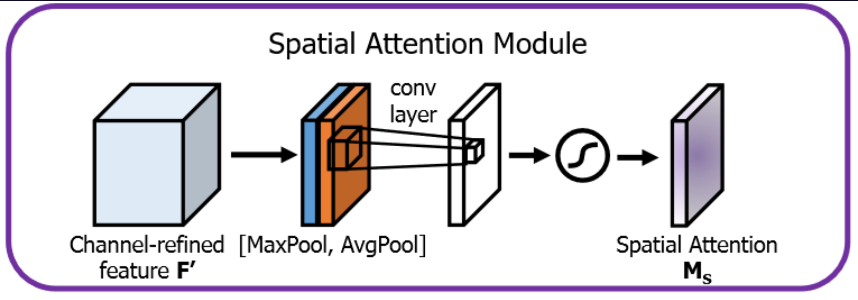
Coordinate Attention



CBAM - convolutional block attention module





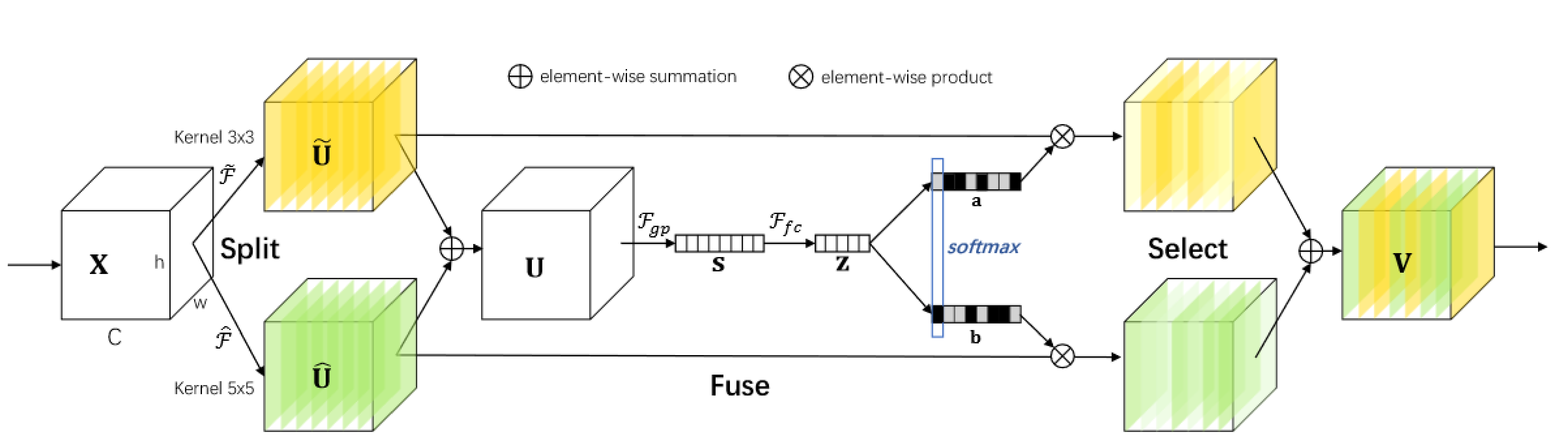


CBAM - convolutional block attention module

Architecture	Param.	GFLOPs	Top-1 Error (%)	Top-5 Error (%)
ResNet18 [5]	11.69M	1.814	29.60	10.55
ResNet18 [5] + SE [28]	11.78M	1.814	29.41	10.22
ResNet18 [5] + CBAM	11.78M	1.815	**29.27**	**10.09**
ResNet34 [5]	21.80M	3.664	26.69	8.60
ResNet34 [5] + SE [28]	21.96M	3.664	26.13	8.35
ResNet34 [5] + CBAM	21.96M	3.665	**25.99**	**8.24**
ResNet50 [5]	25.56M	3.858	24.56	7.50
ResNet50 [5] + SE [28]	28.09M	3.860	23.14	6.70
ResNet50 [5] + CBAM	28.09M	3.864	**22.66**	**6.31**
ResNet101 [5]	44.55M	7.570	23.38	6.88
ResNet101 [5] + SE [28]	49.33M	7.575	22.35	6.19
ResNet101 [5] + CBAM	49.33M	7.581	**21.51**	**5.69**
WideResNet18 [6] (widen=1.5)	25.88M	3.866	26.85	8.88
WideResNet18 [6] (widen=1.5) + SE [28]	26.07M	3.867	26.21	8.47
WideResNet18 [6] (widen=1.5) + CBAM	26.08M	3.868	**26.10**	**8.43**
WideResNet18 [6] (widen=2.0)	45.62M	6.696	25.63	8.20
WideResNet18 [6] (widen=2.0) + SE [28]	45.97M	6.696	24.93	7.65
WideResNet18 [6] (widen=2.0) + CBAM	45.97M	6.697	**24.84**	**7.63**
ResNeXt50 [7] (32x4d)	25.03M	3.768	22.85	6.48
ResNeXt50 [7] (32x4d) + SE [28]	27.56M	3.771	**21.91**	6.04
ResNeXt50 [7] (32x4d) + CBAM	27.56M	3.774	21.92	**5.91**
ResNeXt101 [7] (32x4d)	44.18M	7.508	21.54	5.75
ResNeXt101 [7] (32x4d) + SE [28]	48.96M	7.512	21.17	5.66
ResNeXt101 [7] (32x4d) + CBAM	48.96M	7.519	**21.07**	**5.59**

* all results are reproduced in the PyTorch framework.

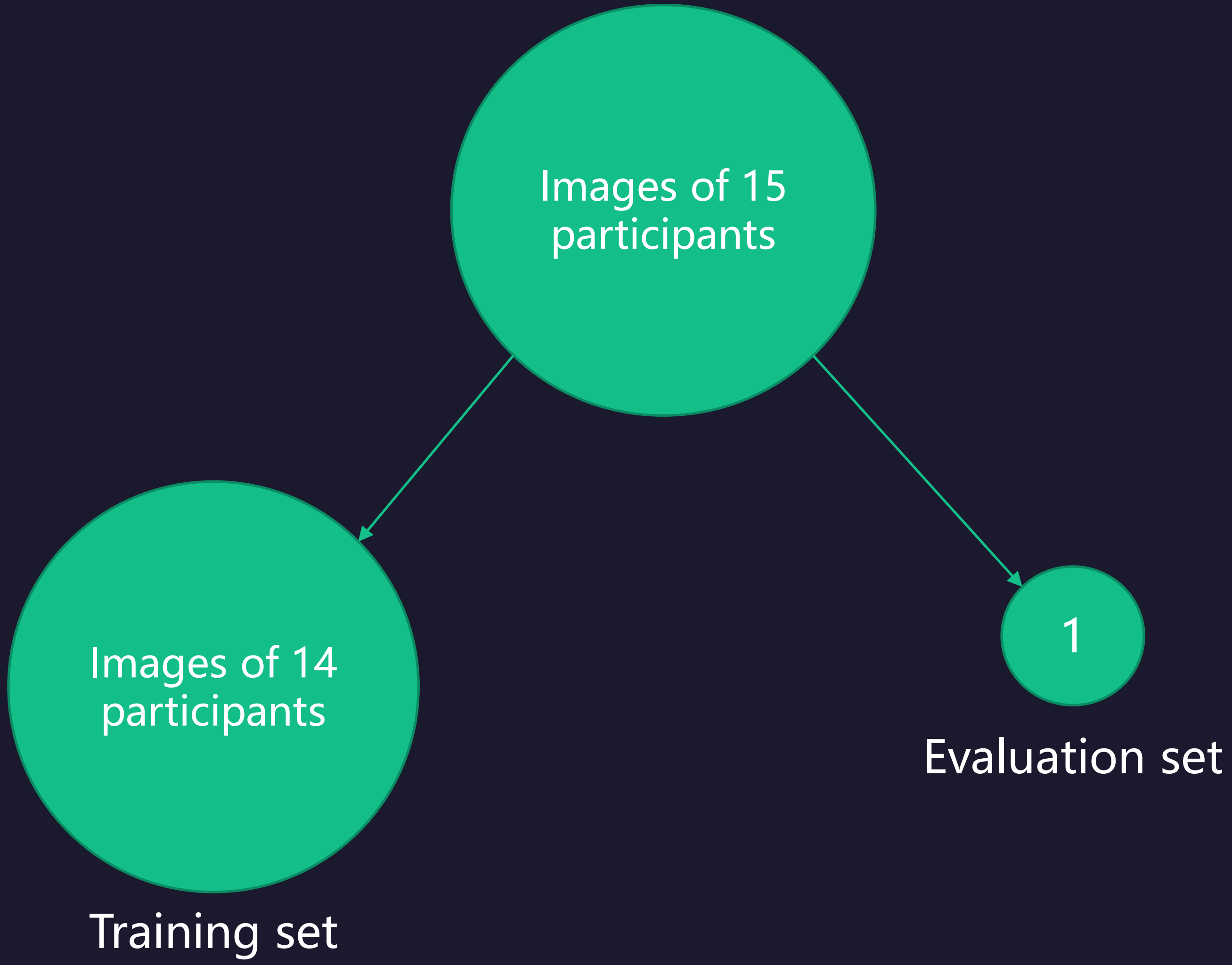
SK-Net


element-wise summation
element-wise product
Kernel 3x3
$\tilde{\mathcal{F}}$
$\widetilde{\mathbf{U}}$
X
h
w
C
Split
$\hat{\mathcal{F}}$
Kernel 5x5
$\widehat{\mathbf{U}}$
U
$\mathcal{F}_{gp}$
s
$\mathcal{F}_{fc}$
z
a
softmax
b
Fuse
Select
V

SK-Net

	top-1 err (%)		#P	GFLOPs
	224×	320×		
ResNeXt-50	22.23	21.05	25.0M	4.24
AttentionNeXt-56 [44]	21.76	–	31.9M	6.32
InceptionV3 [43]	–	21.20	27.1M	5.73
ResNeXt-50 + BAM [32]	21.70	20.15	25.4M	4.31
ResNeXt-50 + CBAM [45]	21.40	20.38	27.7M	4.25
SENet-50 [12]	21.12	19.71	27.7M	4.25
SKNet-50 (ours)	**20.79**	**19.32**	27.5M	4.47
ResNeXt-101	21.11	19.86	44.3M	7.99
Attention-92 [44]	–	19.50	51.3M	10.43
DPN-92 [5]	20.70	19.30	37.7M	6.50
DPN-98 [5]	20.20	18.90	61.6M	11.70
InceptionV4 [41]	–	20.00	42.0M	12.31
Inception-ResNetV2 [41]	–	19.90	55.0M	13.22
ResNeXt-101 + BAM [32]	20.67	19.15	44.6M	8.05
ResNeXt-101 + CBAM [45]	20.60	19.42	49.2M	8.00
SENet-101 [12]	20.58	18.61	49.2M	8.00
SKNet-101 (ours)	**20.19**	**18.40**	48.9M	8.46

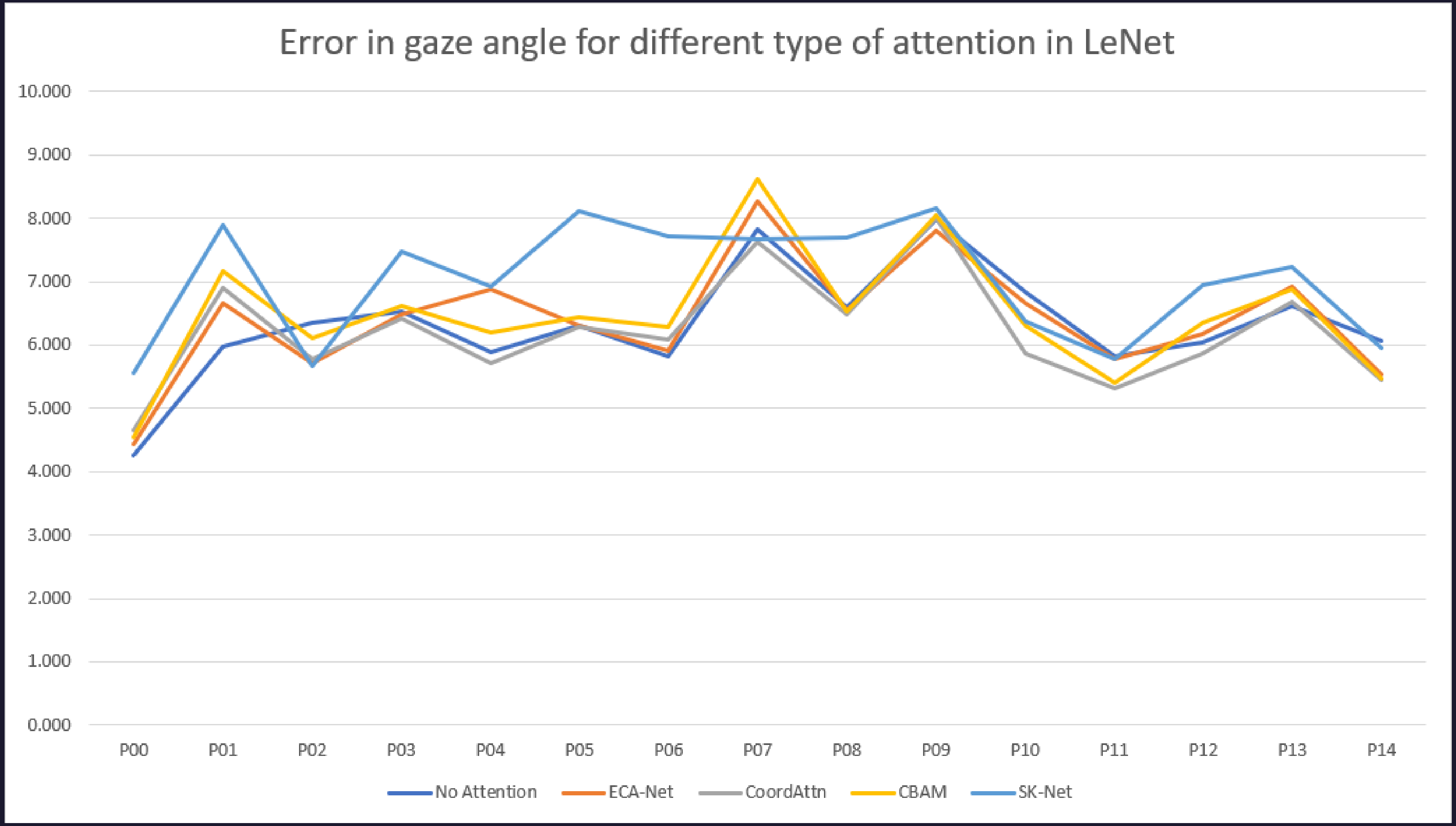
Experiments



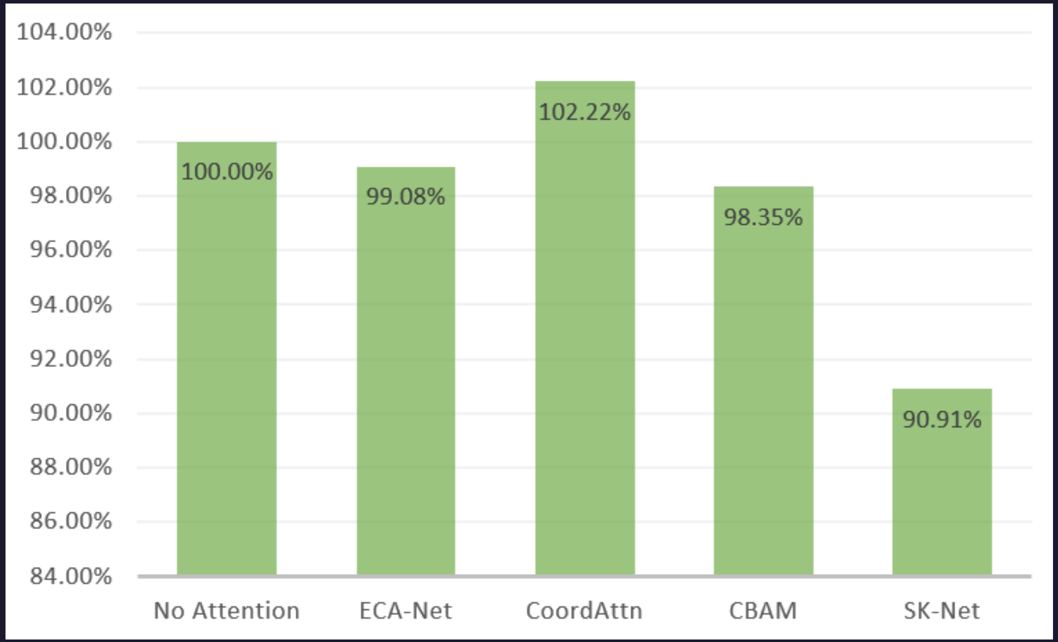
Experiments – Performance Metrics

- Error of estimated gaze angle
- Training time for each epoch

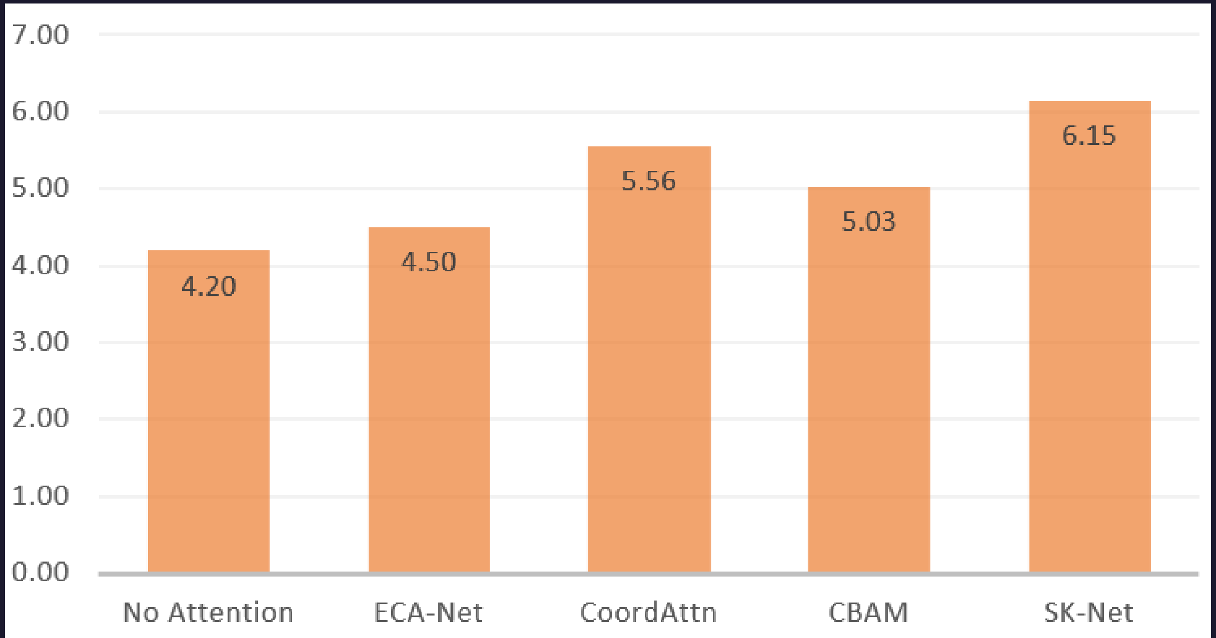
Results – LeNet



Results – LeNet

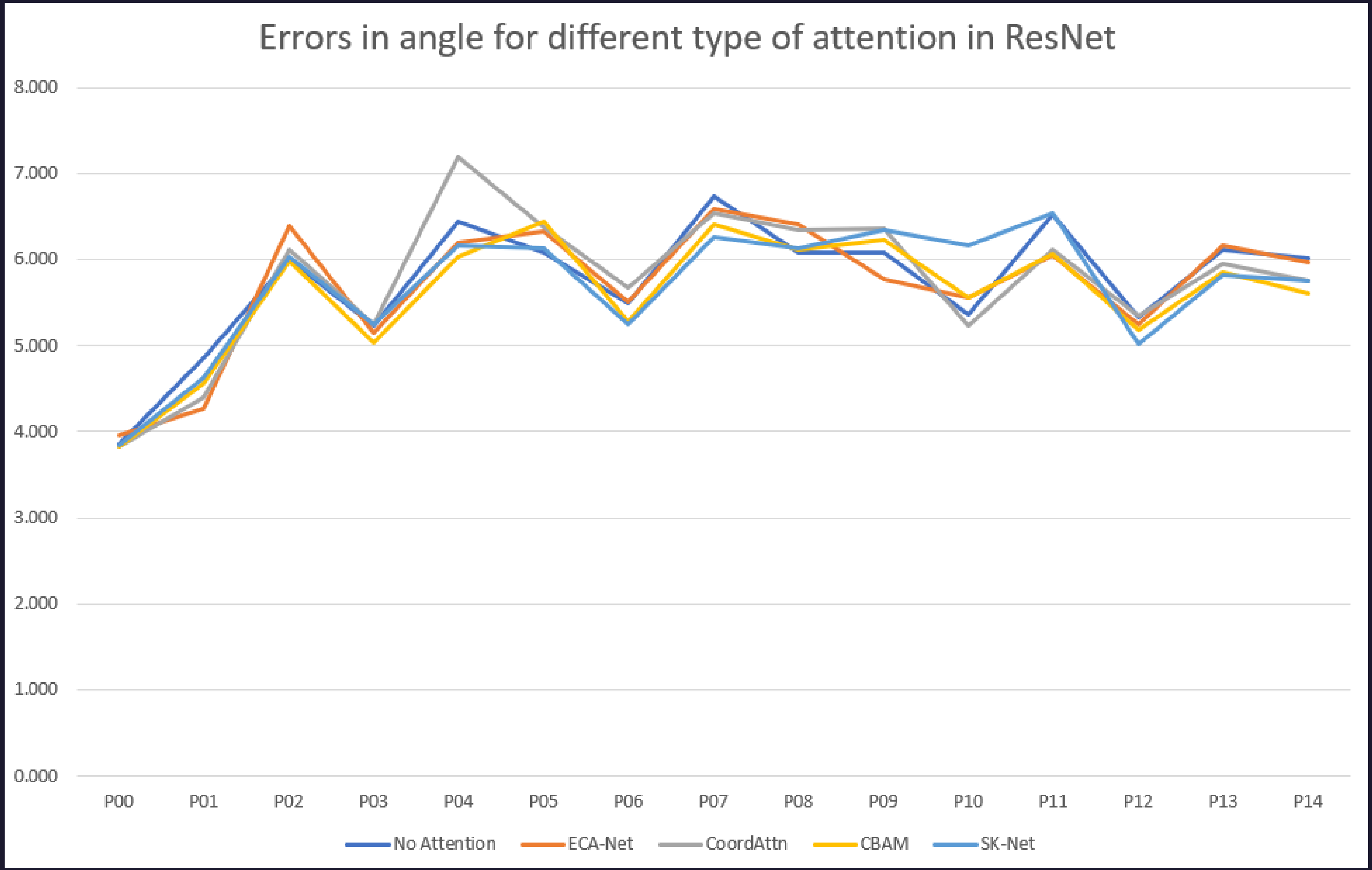


Relative performance in angle error

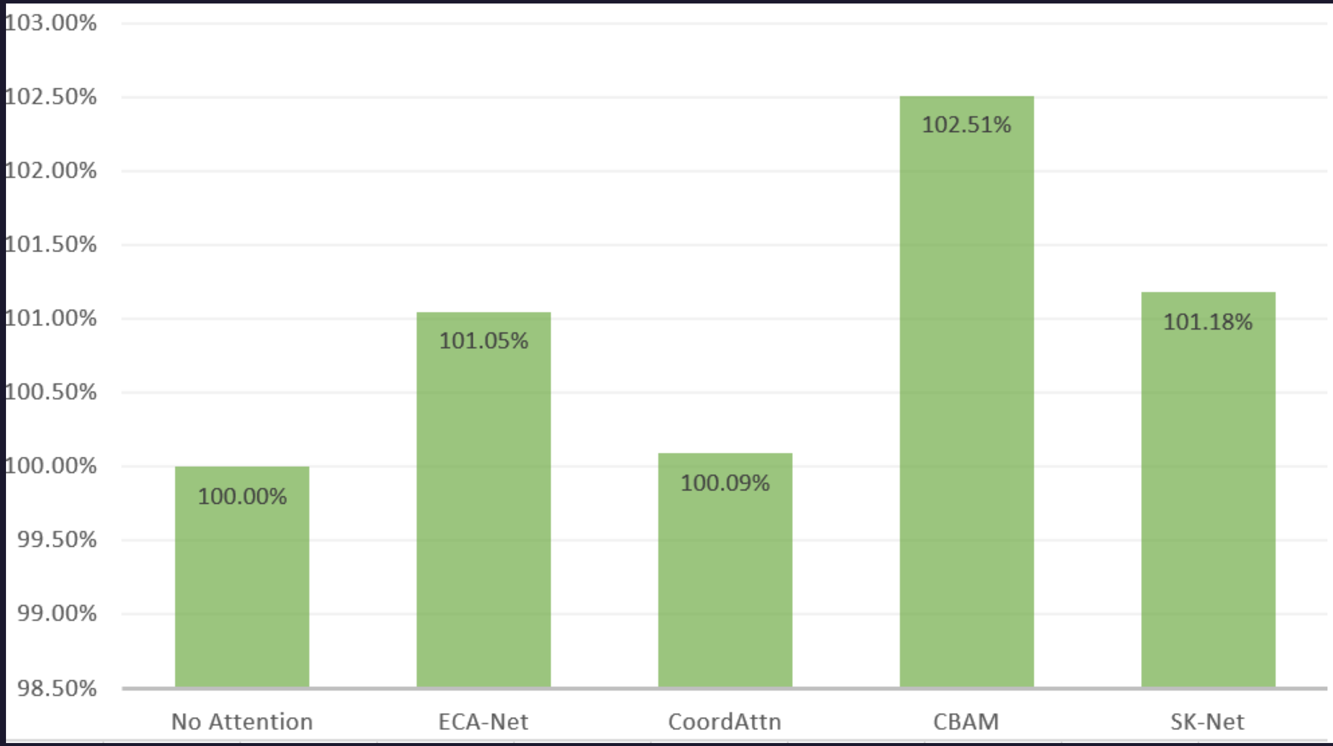


Training time for each epoch

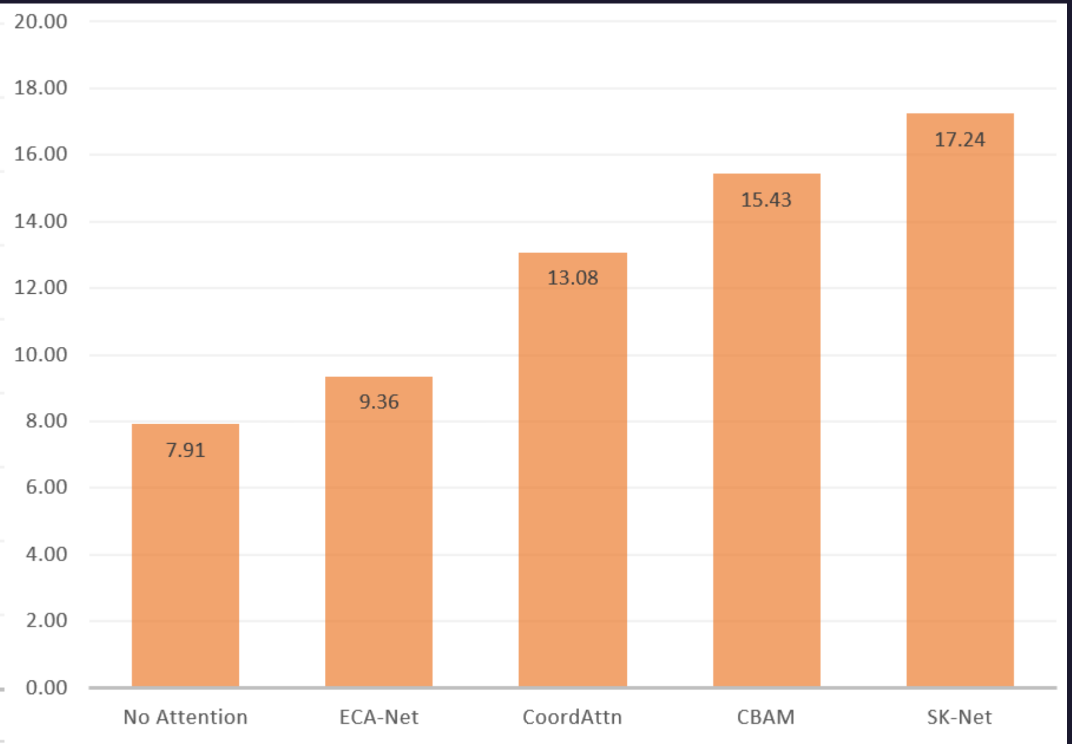
Results – ResNet



Results – ResNet



Relative performance in angle error



Training time for each epoch

Conclusion

- 2.2% better with coordinate attention in LeNet
- 2.5% better with CBAM in ResNet
- Higher time consuming in training

Future Works

- Fine tuning parameters of attention layers
- Combine the attention layer with calibration

Thank you for listening !

Reference

- [Tobii, 2011] Tobii. Accuracy and precision test method for remote eye trackers. Stockholm, Sweden, 2011.
- [Xu et al., 2015] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhutdinov, Richard Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention, 2015.
- [Lecun et al., 1998] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. Proceedings of the IEEE, 86(11):2278–2324, 1998.
- [He et al., 2015] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. CoRR, abs/1512.03385, 2015.
- [Wang et al., 2019] Qilong Wang, Banggu Wu, Pengfei Zhu, Peihua Li, Wangmeng Zuo, and Qinghua Hu. Eca-net: Efficient channel attention for deep convolutional neural networks. CoRR, abs/1910.03151, 2019.
- [Hou et al., 2021] Qibin Hou, Daquan Zhou, and Jiashi Feng. Coordinate attention for efficient mobile network design. CoRR, abs/2103.02907, 2021.
- [Woo et al., 2018] Sanghyun Woo, Jongchan Park, Joon- Young Lee, and In So Kweon. CBAM: convolutional block attention module. CoRR, abs/1807.06521, 2018.
- [Li et al., 2019] Xiang Li, Wenhai Wang, Xiaolin Hu, and Jian Yang. Selective kernel networks. CoRR, abs/1903.06586, 2019.