NANYANG TECHNOLOGICAL UNIVERSITY
SINGAPORE

School of Electrical and Electronic Engineering

DRIVER ACTION RECOGNITION USING ARTIFICIAL INTELLIGENCE

Name: Philip Lee Hann Yung
(U1923382D)

Supervisor: Assoc Prof Yap Kim Hui
Examiner: Prof Wen Changyun

Date: 9 May 2023

Agenda

BACKGROUND

LITERATURE REVIEW

METHODOLOGY

RESULTS

CONCLUSION

Motivation

Background

Distracted Driving

Background

Any activity that diverts attention from driving

2019 US: 1162 injured + 9 killed / day

Reduce distracted driving!

Solution:
Cabin Monitoring

Distracted Driving

[1] *DISTRACTED DRIVERS #1 Crashing while using a Phone* || *Dash Cam Compilation*, (Jun. 11, 2019). Accessed: Apr. 12, 2023. [Online Video]. Available: https://www.youtube.com/watch?v=_pccGggeW7Y

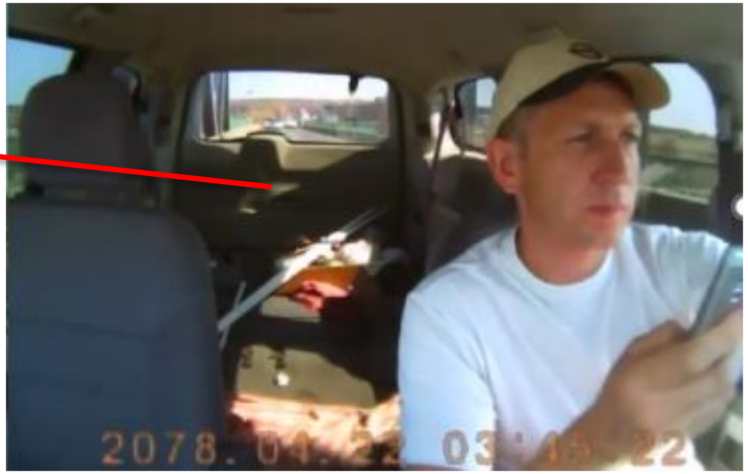
[2] “Distracted Driving | NHTSA.” https://www.nhtsa.gov/risky-driving/distracted-driving (accessed Oct. 32, 2022).

[3] National Center for Statistics and Analysis, *Distracted Driving 2019*, (Research Note. Report No. DOT HS 813 111). National Highway Traffic Safety Administration, 2021.

Real-world Implications

Background

- Cabin-monitoring system
  - May be mandated for truck/bus drivers
  - Safety for road users
- E.g. Using mobile phone earlier!



Objective & Scope

- Train deep learning-based models which can recognize driver’s actions in a car cabin environment.
  - Robust, Real-time, Action after is out-of-scope

1. Survey datasets and models
2. Train and evaluate models
3. Improve model robustness
4. Construct dataset and fine-tune

LITERATURE REVIEW

Literature Review

Video Action Recognition

Requires: Spatio-temporal modelling capability

2015 3D-CNNs

2017 Kinetics400
Inflated-3D ConvNet (I3D)

Hand-crafted features → Two-stream networks → 3D-CNNs → Efficient video modelling

2019 Temporal Shift Module (TSM)

[5] Y. Zhu *et al.*, “A Comprehensive Study of Deep Video Action Recognition.” arXiv, Dec. 11, 2020. doi: 10.48550/arXiv.2012.06567.

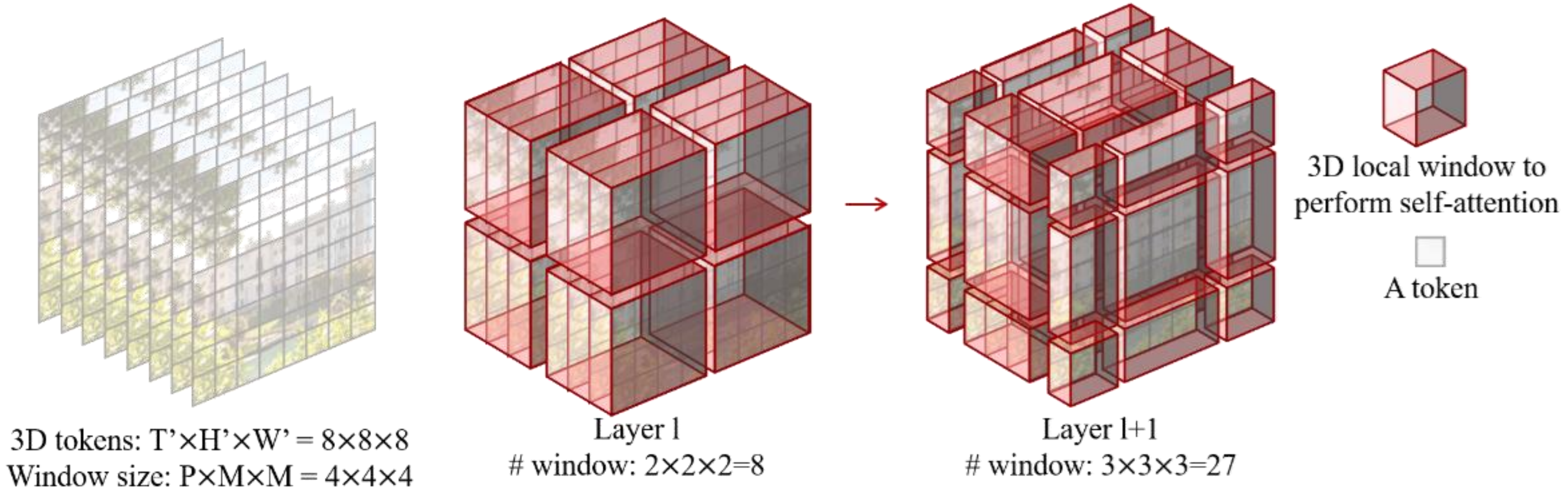
Video Swin Transformer (VST)

2015 3D-CNNs

2017 Kinetics400
Inflated-3D ConvNet (I3D)

2019 Temporal Shift Module (TSM)

2022 Video Swin Transformer (VST)



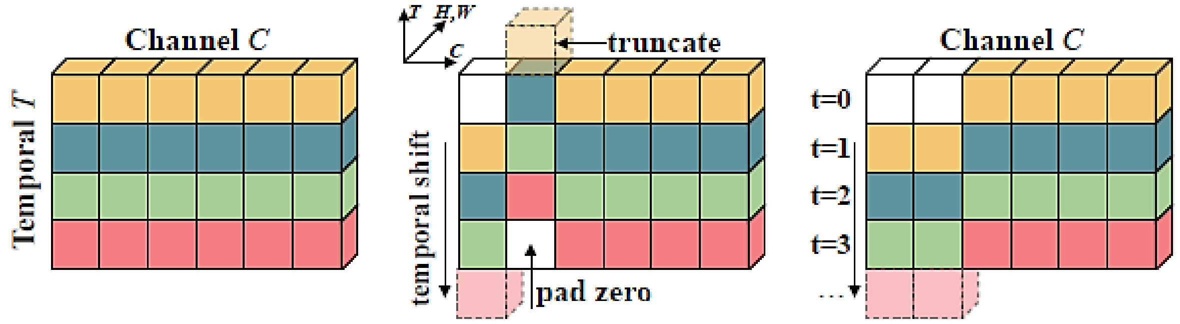
[6] Z. Liu *et al.*, “Video Swin Transformer,” in *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2022, pp. 3192–3201. doi: 10.1109/CVPR52688.2022.00320.

Temporal Shift Module (TSM)

Literature Review

Main Idea:

- Shift along temporal dimension to exchange information.
- Must not harm spatial feature learning capability.



(a) The original tensor without shift. (b) Offline temporal shift (bi-direction). (c) Online temporal shift (uni-direction).

Enjoys 2D-CNN efficiencies

- ❑ Optimised hardware implementation
- ❑ Availability of pretrained weights

[7] J. Lin, C. Gan, and S. Han, “TSM: Temporal Shift Module for Efficient Video Understanding,” in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, Oct. 2019, pp. 7082–7092. doi: 10.1109/ICCV.2019.00718

Dataset

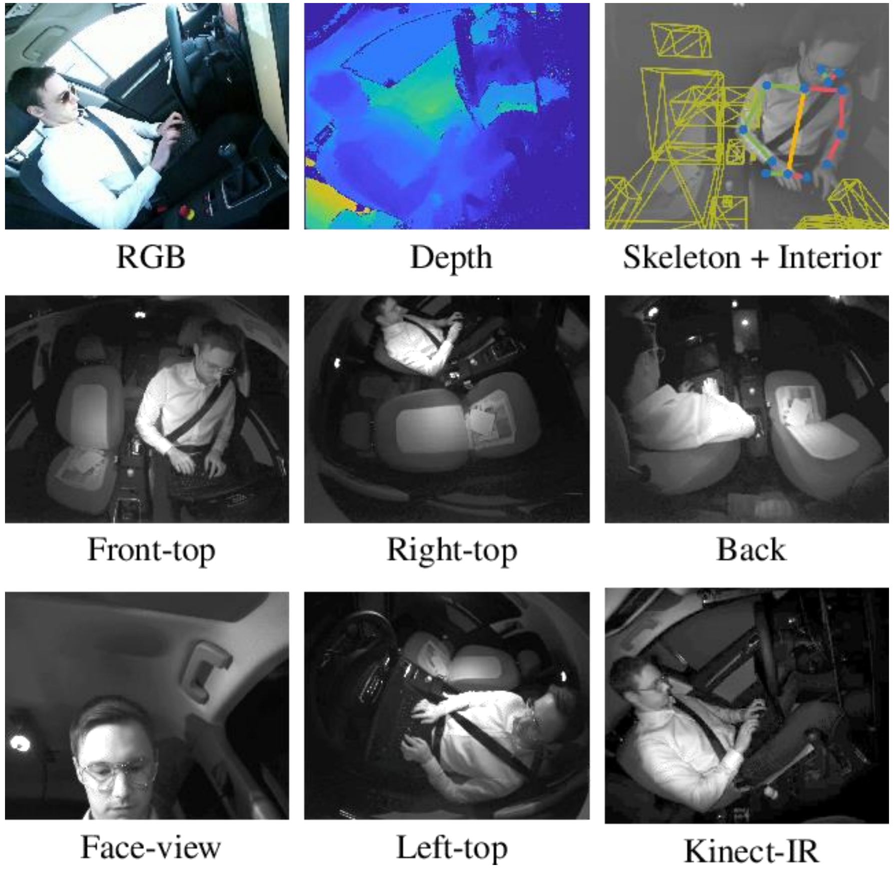
Drive&Act (DAA), 2019

- Multi-modal, multi-view
- 83 activity labels (3 levels)
- 3 train-val-test splits
- 15 participants for variability

Largest dataset: 9.6m frames!

(in car-cabin activity domain)

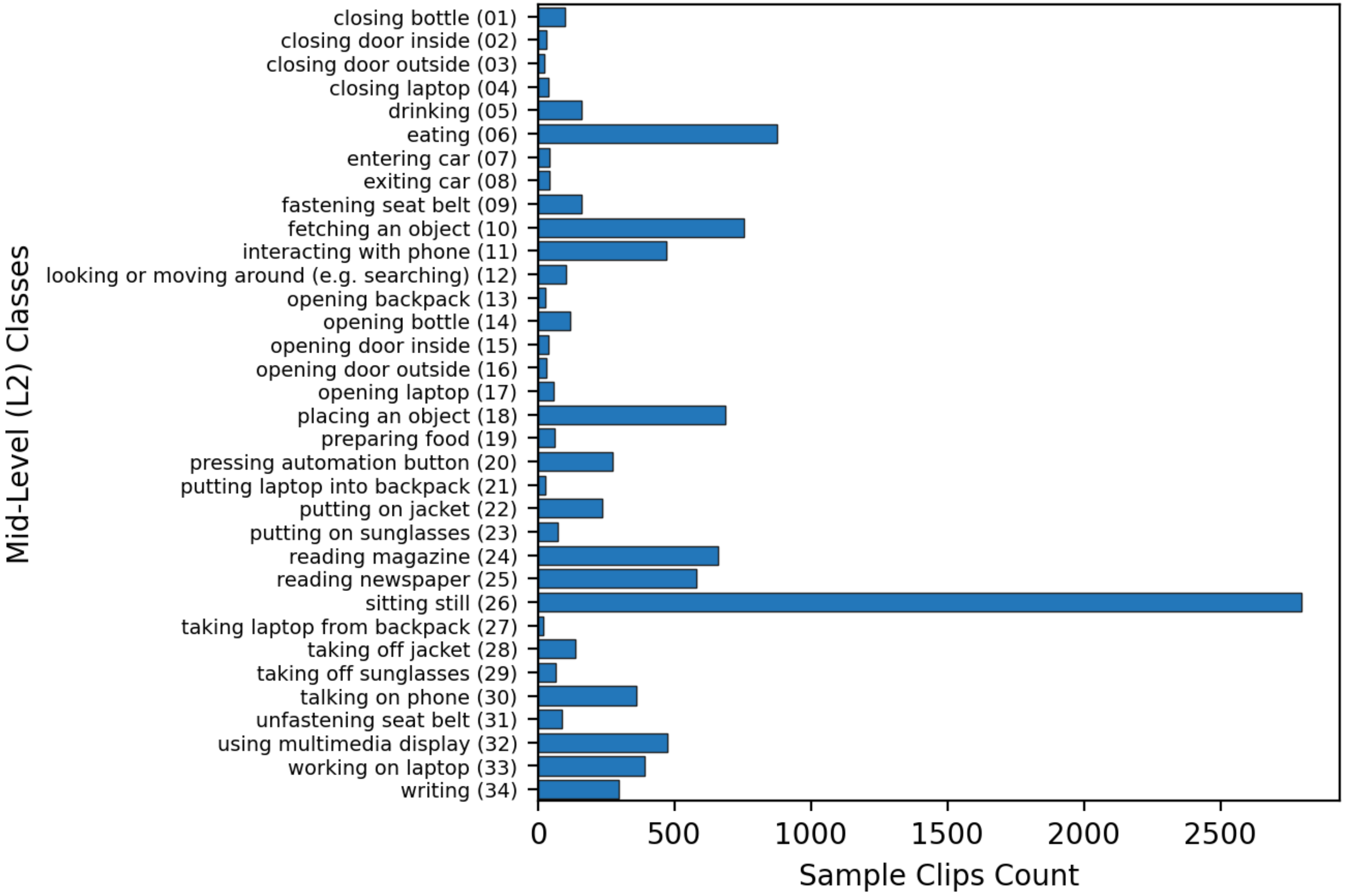
TICaM: A Time-of-flight In-car Cabin Monitoring (123k frames)



[8] M. Martin *et al.*, "Drive&Act: A Multi-Modal Dataset for Fine-Grained Driver Behavior Recognition in Autonomous Vehicles," in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, Oct. 2019, pp. 2801–2810. doi: 10.1109/ICCV.2019.00289

Metric

Literature Review



<u>BalAcc Score</u>

Mean class accuracy

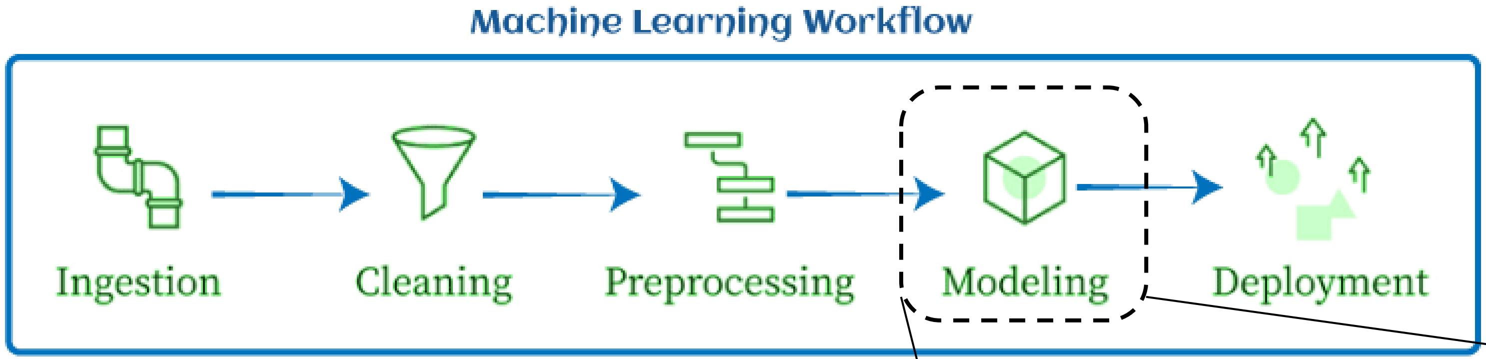
Class imbalance issue

$$\frac{1}{N_c}\sum_{c=1}^{N_c}\frac{TP_c}{TP_c+FN_c}$$

Methodology, Results

SINGLE-MODALITY

End-to-end ML Project Pipeline



Live Real-time System

Fine-tune on Conti-Action-2023

30 FPS Live System

Dataset

Drive&Act

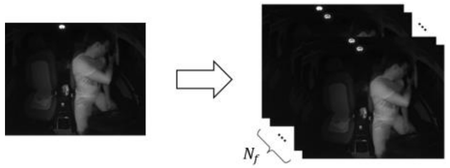
Conti-Action-2023

Pre-processing

Video → Raw frames

Normalisation

Resize and Crop



1 Train

3 model families (TSM, I3D, VST)

2 Fine-tune

Hyperparameters

3 Evaluate

Validation and testing sets (BalAcc score)

4 Improve

Class imbalance mitigation

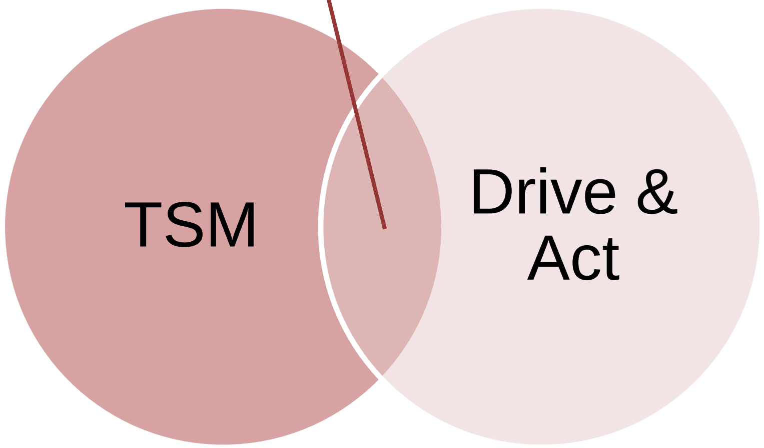
Multi-modal fusion

[9] “Machine Learning Pipeline - Javatpoint,” *www.javatpoint.com*. https://www.javatpoint.com/machine-learning-pipeline (accessed May 08, 2023).

① Training

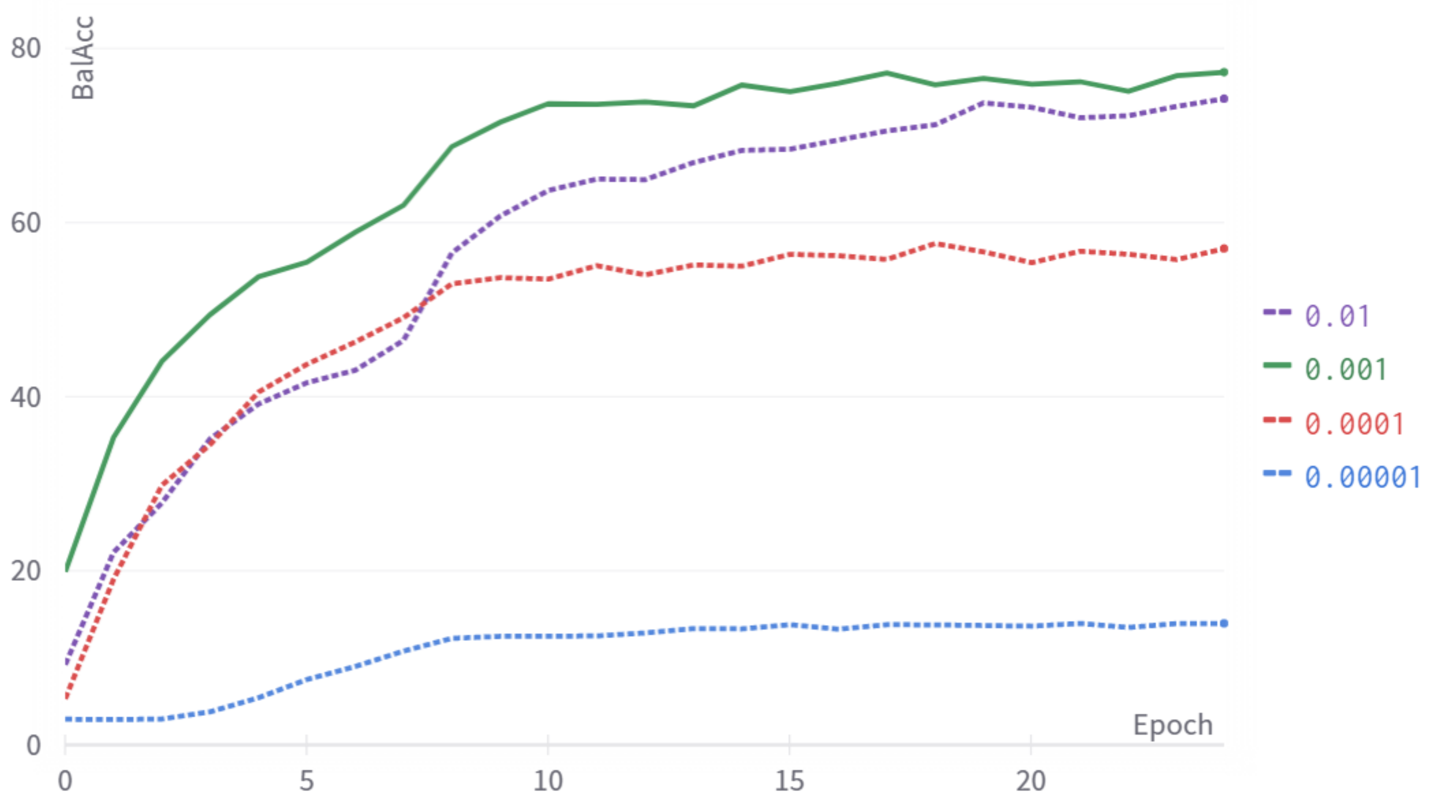
- 3 model families
  - TSM (main method)
  - I3D
  - VST
- 2 datasets
  - Drive&Act (benchmark)
  - Conti-Action-2023 (demo only)

Research Gap!



② Varying Learning Rate

Methodology



Sample BalAcc curve for selecting learning rate hyperparameter

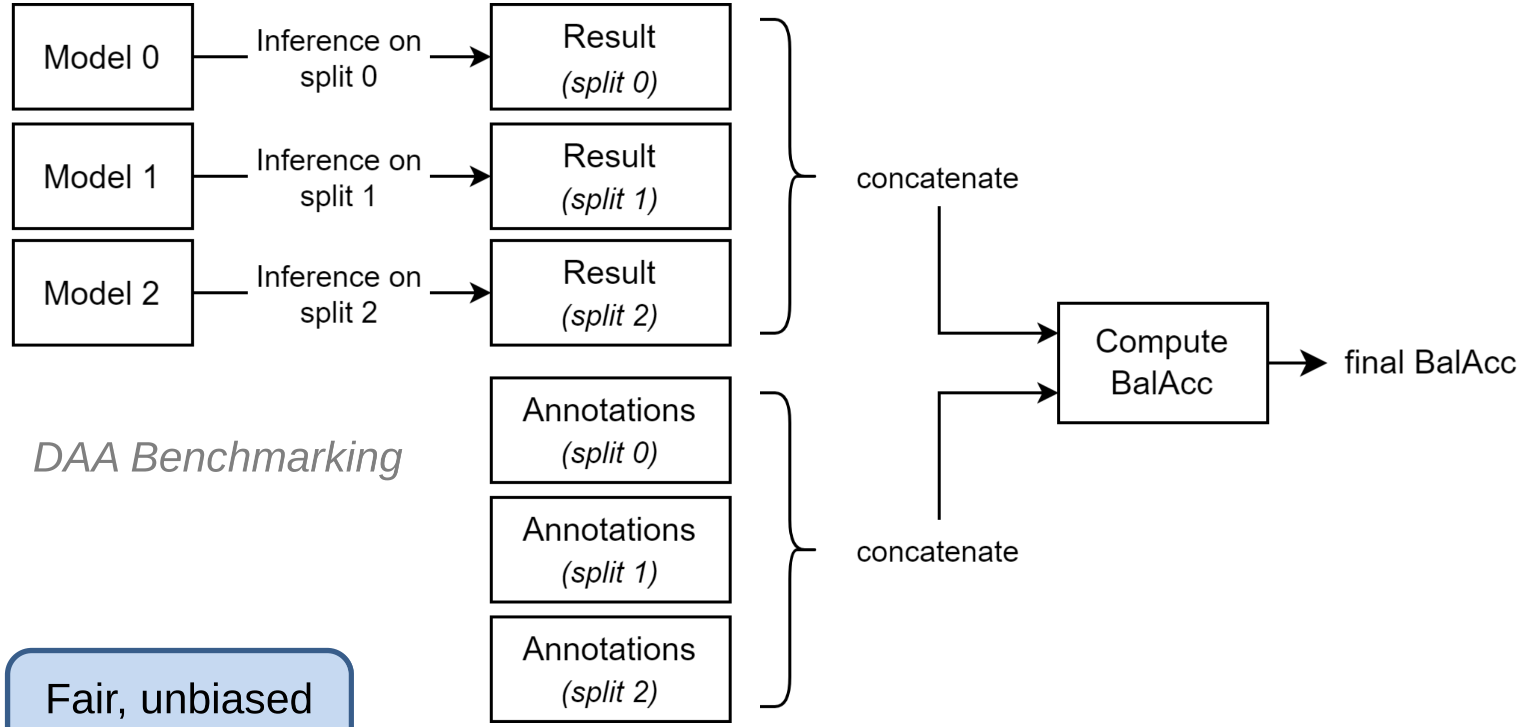
② Fine-tune Hyperparameters

Training Parameter	TSM	I3D	VST
N_f (No. of Frames)	8, 16	32	32
Backbone	ResNet50	ResNet50	VST: Tiny, Base
Pretrained Weights	ImageNet/Kinetics400	Kinetics400	Kinetics400
Batch Size	32, 16	128	32, 16
Number of Epochs	25	20	20
Learning Rate	0.0010, 0.0005	0.1000	0.0004
LR Scheduler	Step decay: 9, 17	Step decay: 9, 16	Cosine annealing
Optimizer	SGD	SGD	AdamW

Loss function: Cross-entropy loss

③ Evaluation Methodology

Spatial Crop



DAA Benchmarking

Fair, unbiased comparison!

④ Class Imbalance Mitigation

- Class Weighting
- Uniform Class Sampling
- Hard Sample Mining
- CW + HSM

Modified cross-entropy loss function

$$w_c \propto \frac{1}{f_c}; \quad \mathrm{CE}_i = -\sum_{c=1}^{N_c} w_c y_{i,c} \log p_{i,c}$$

```
Hard Sample Mining Algorithm

after every 3 epoch:
    evaluate loss for each training sample
    hard samples = (sample loss) > 1.2 * (train loss mean)
    train model on hard samples
```

Results

Results: Single Modality

- TSM is recommended for real-time use applications!

Model	Ns	Val ↑	Test ↑	Latency ↓	Throughput ↑
I3D	32	63.29	60.41	18.3 ms	54.7 clips/s
TSM	8	68.47	62.34	**15.0 ms**	**66.7 clips/s**
VST: Tiny	32	68.67	60.10	40.2 ms	24.9 clips/s
VST: Base	32	**70.06**	**63.48**	89.3 ms	11.2 clips/s

Results are best from each model family, training procedure already includes class imbalance mitigation techniques.

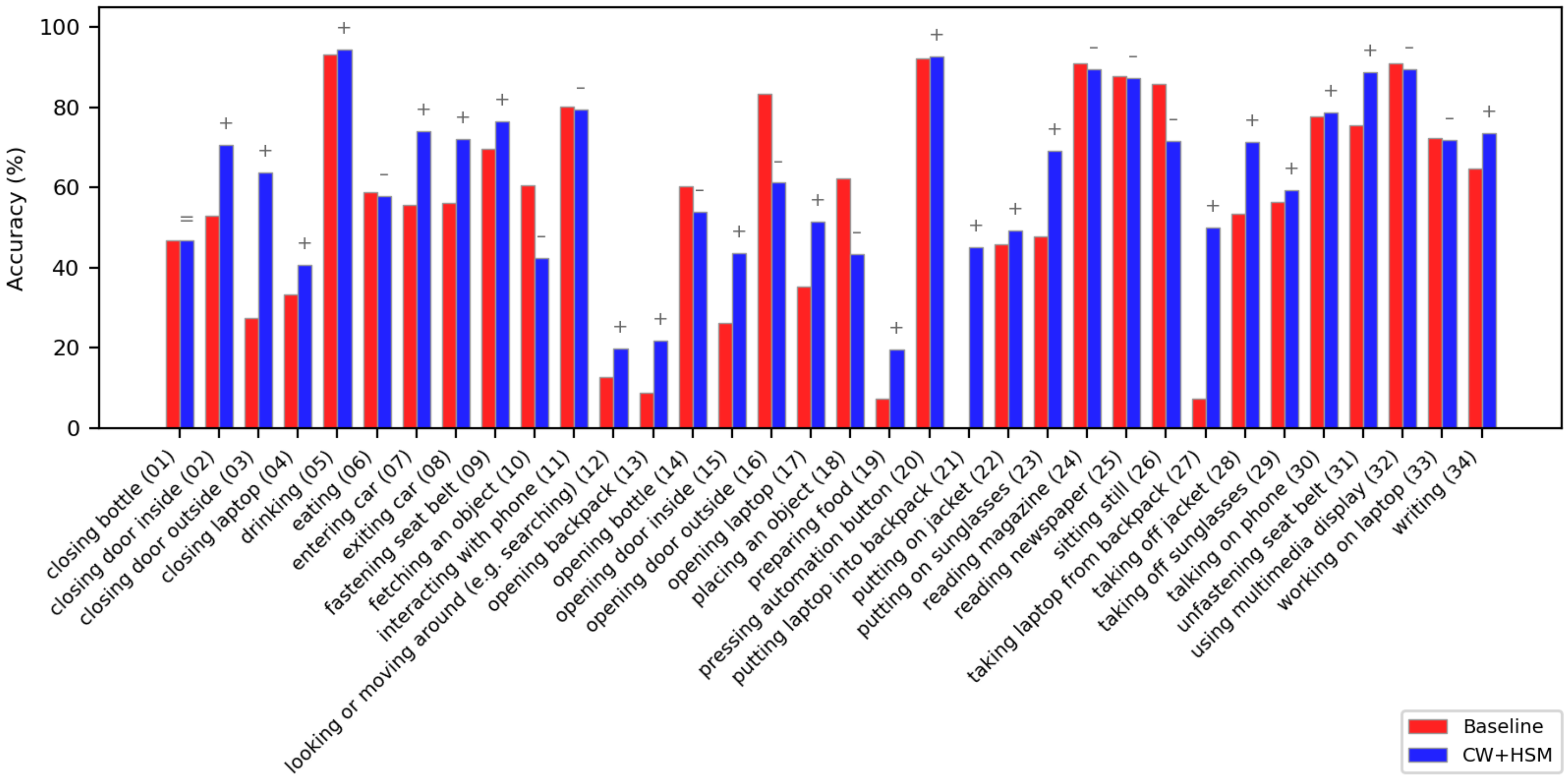
TSM: CW+HSM *I3D, VST: Uniform Class Sampling*

Results: Single Modality

Results

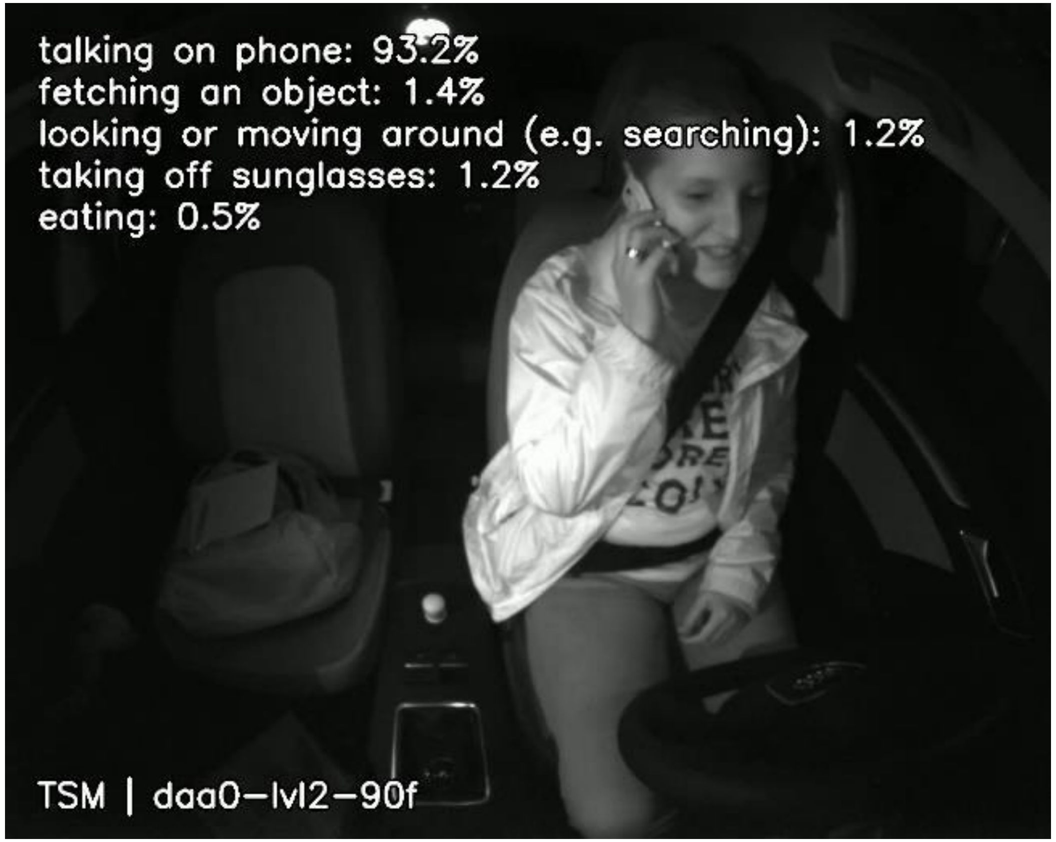
TSM:
Base to CW+HSM

22 out of 34 classes ↑
(60.17 → 68.47)



Results: Single Modality

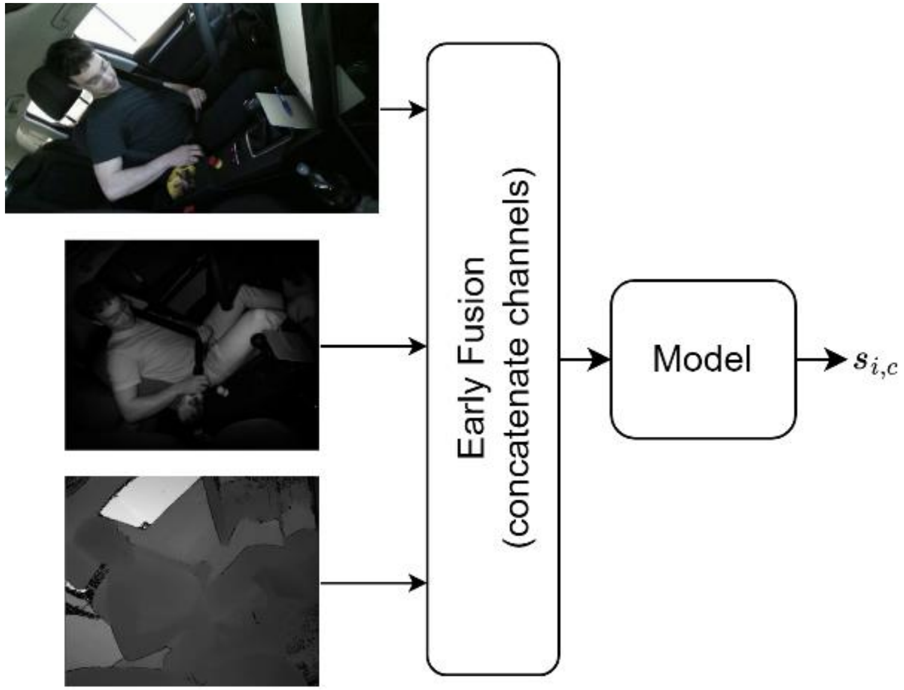
Results



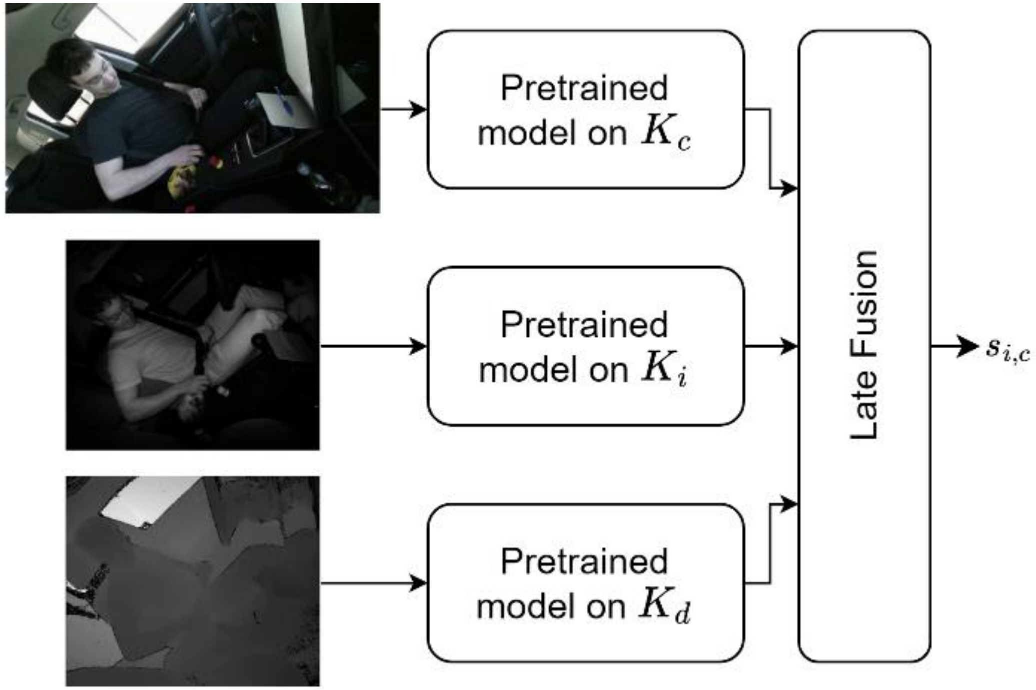
Methodology, Results

MULTI-MODALITY

④ Multi-modality Fusion

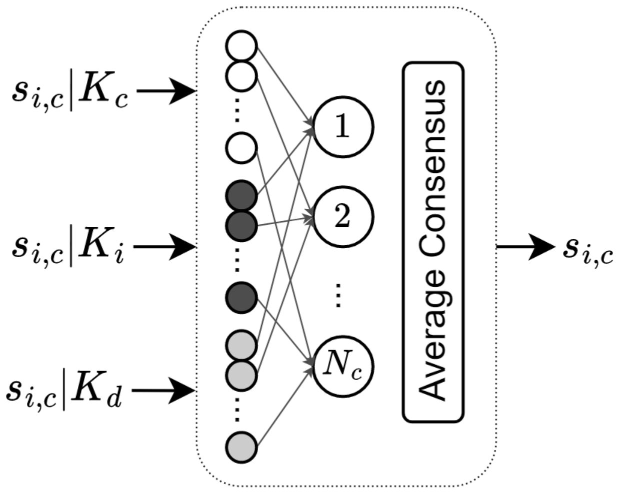


1 architecture



3 proposed architectures

Results: Multi-modality



(C) *Learned weights*

$$s_{i,c} = \sum_{m=1}^{N_m} w_{m,c} s_{i,c,m}$$

Fusion Method	**Modality**	**Val ↑**	**Test ↑**
No Fusion (single modality)	K_c	70.35	62.72
	K_i	69.33	59.81
	K_d	68.31	58.28
Late Fusion: Type C (linear weighted scores)	$K_c + K_i$	71.47	64.99
	$K_c + K_d$	74.06	65.09
	$K_i + K_d$	74.02	63.24
	$K_c + K_i + K_d$	**75.20**	**66.32**

Fusion using TSM model, only best fusion method is shown.

Results: Multi-modality

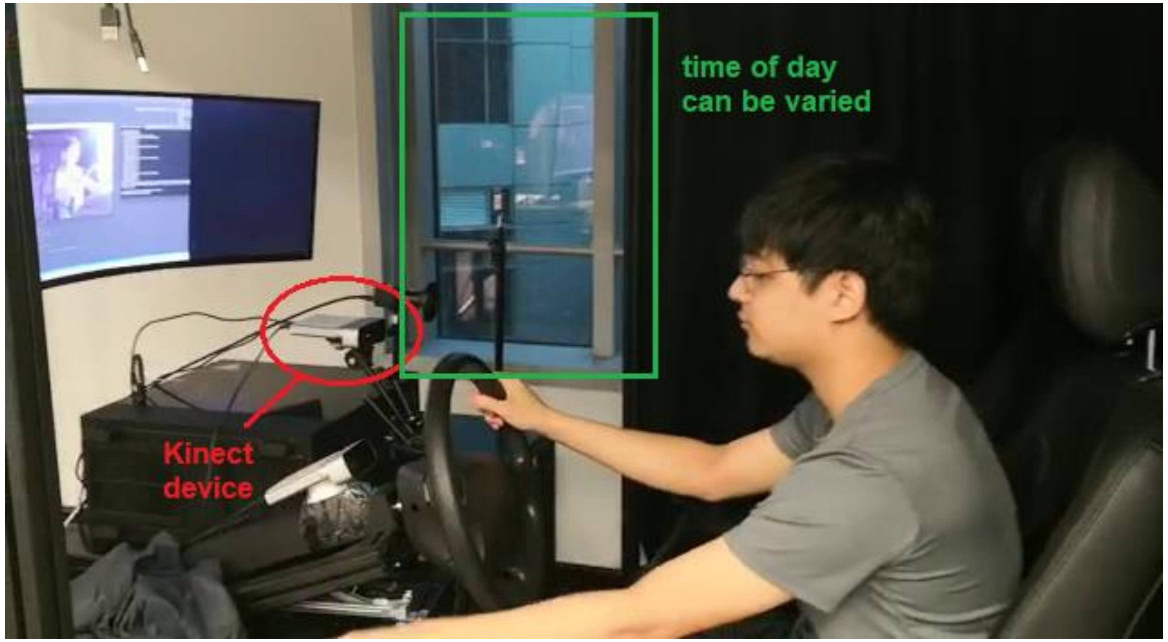
Methodology, Results

LIVE REAL-TIME SYSTEM

Conti-Action-2023

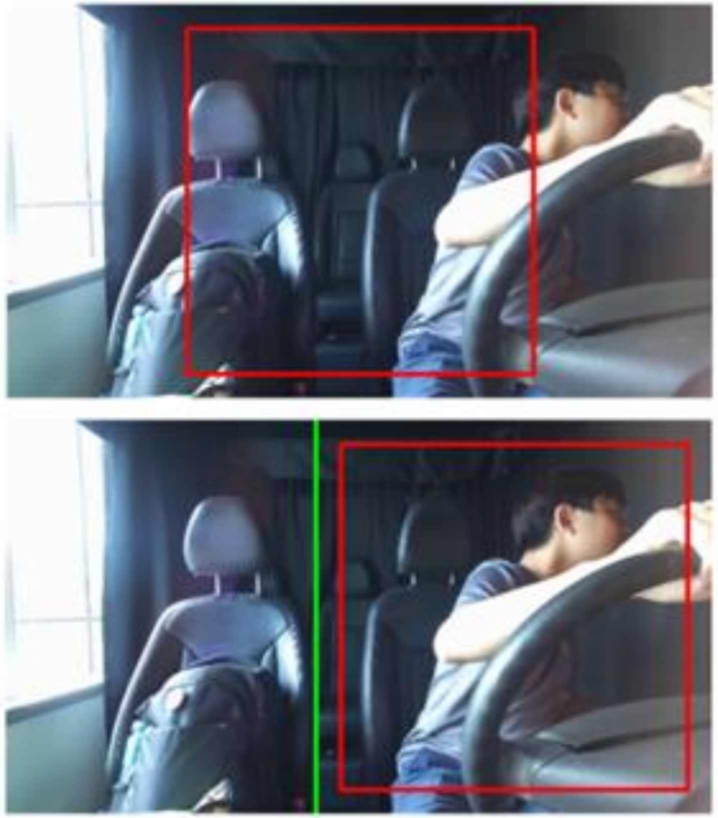
- 15 volunteers
- 32 classes
- Day and night scenes

- Camera position:
  - Center of Dashboard

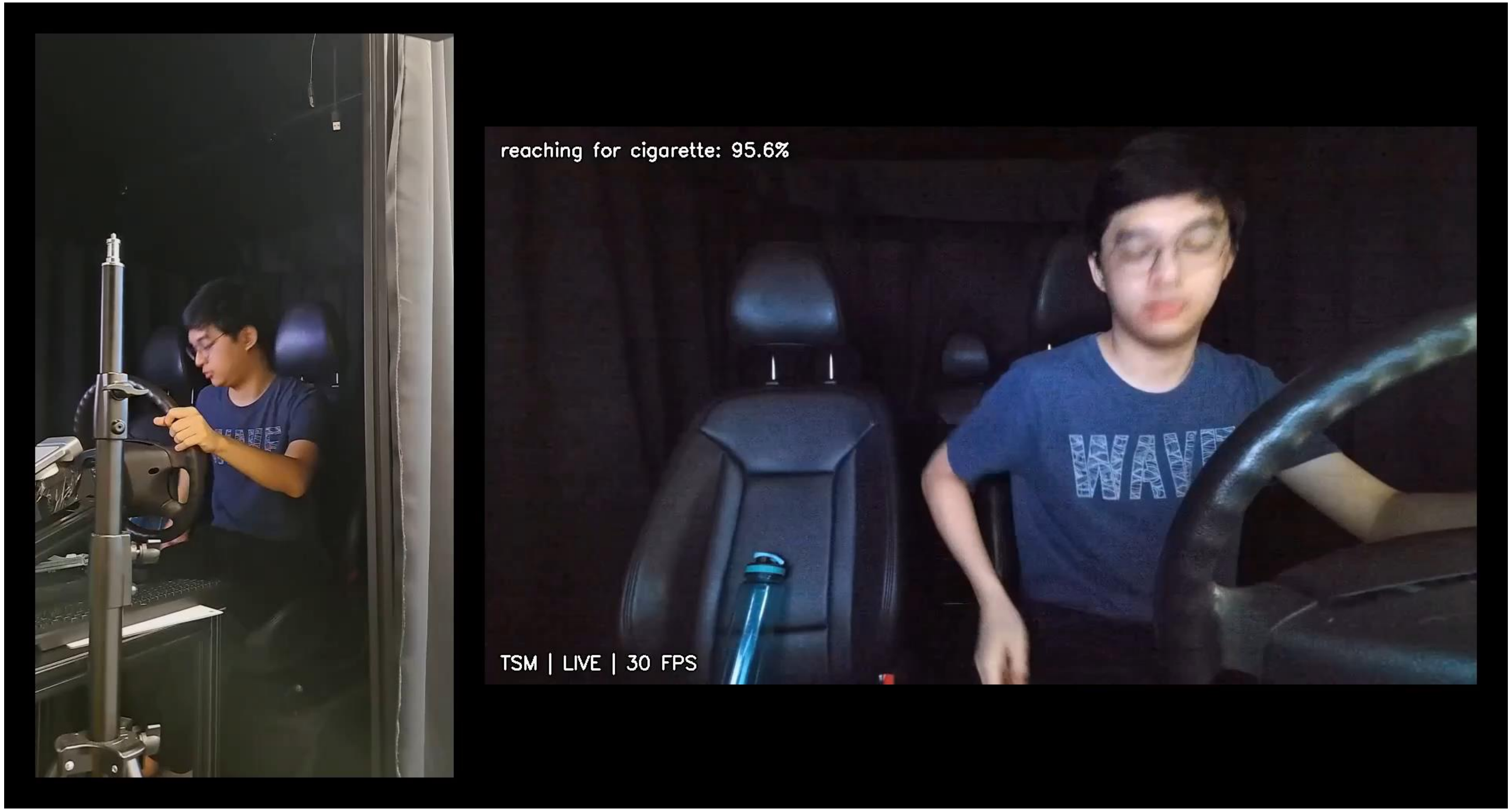


System Optimizations

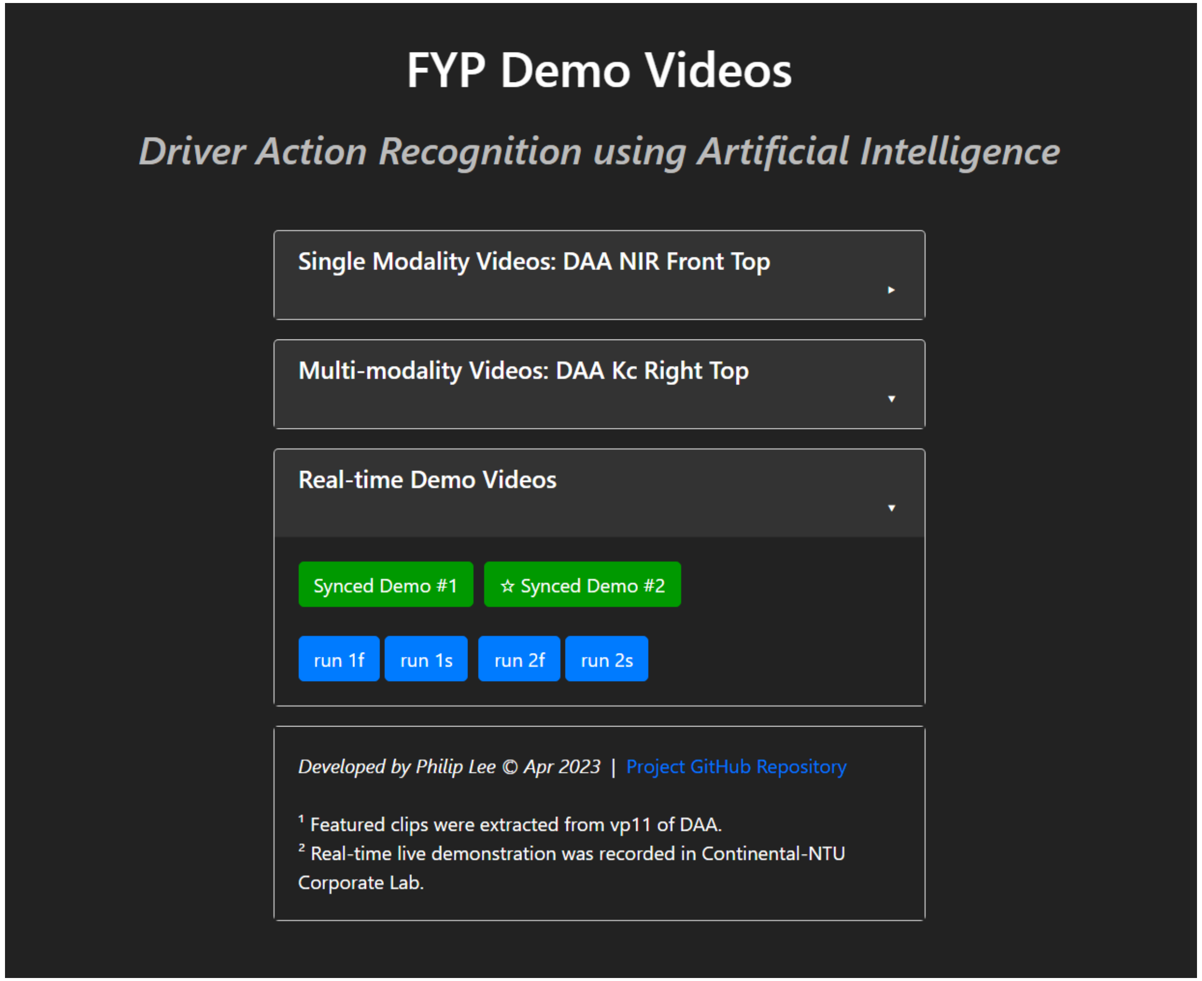
- Right crop then square crop
- Data transformations
  - 94.1 ms → 15.6 ms
  - 6 times improvement



Results: Live Real-time Prototype



Results: Demo Videos



Local webpage for self-exploration

CONCLUSION

Conclusion

Drive&Act (DAA)

Future Work

- More extensive dataset
- Advanced multi-modality fusion techniques
  - Continuation of FYP

Thank you!

QnA

APPENDIX

Background

Appendix

- The number of fatalities in distraction-affected crashes, i.e., a crash involving at least one driver who was distracted, was 3,522, or 8.2 percent of total fatalities in 2021. This represents a 12-percent increase from 3,154 in 2020.

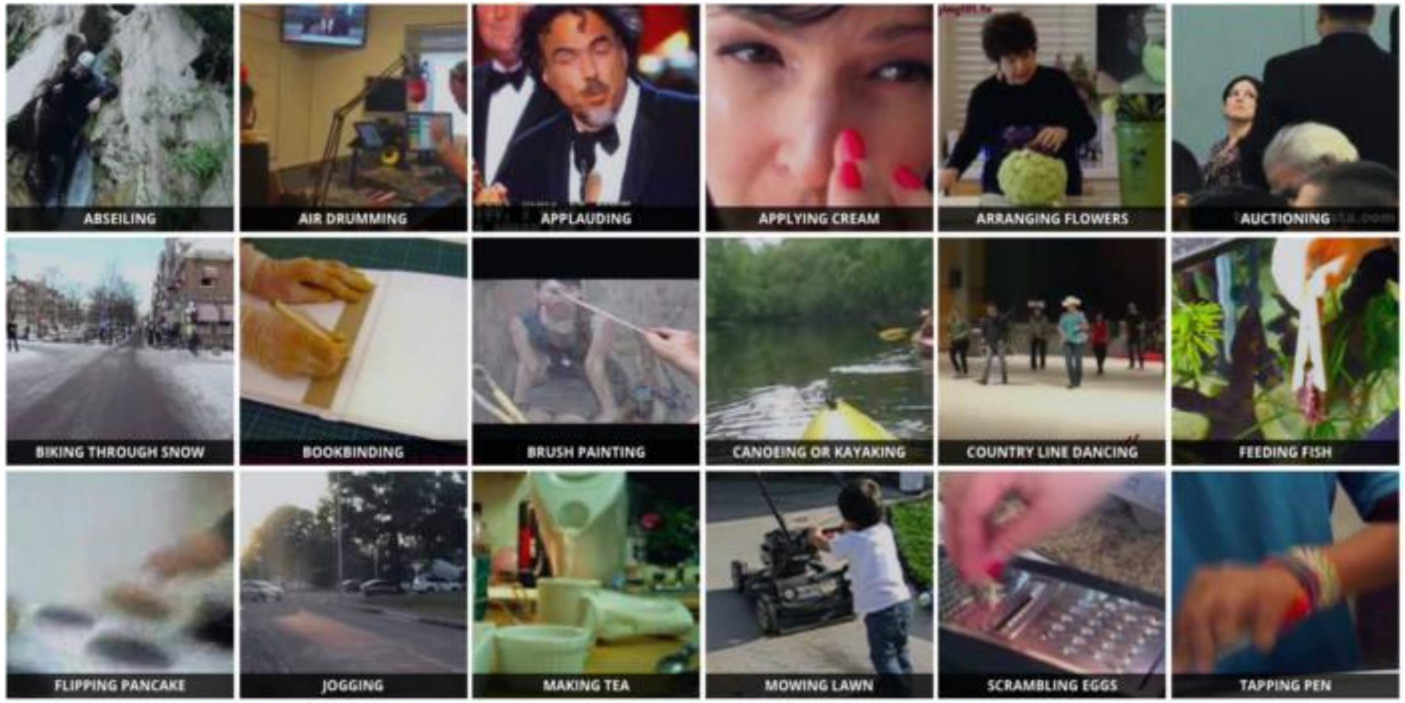
- 3522 / 365 = 9.65 people per year

- https://www.nhtsa.gov/risky-driving/distracted-driving
- https://crashstats.nhtsa.dot.gov/Api/Public/ViewPublication/813435

Kinetics400 Dataset

Appendix

- Realistic action videos
- Collected from YouTube
- 306,245 short trimmed videos
- 400 action categories



W. Kay *et al.*, "The Kinetics Human Action Video Dataset." arXiv, May 19, 2017. doi: 10.48550/arXiv.1705.06950.

Balanced Accuracy Score

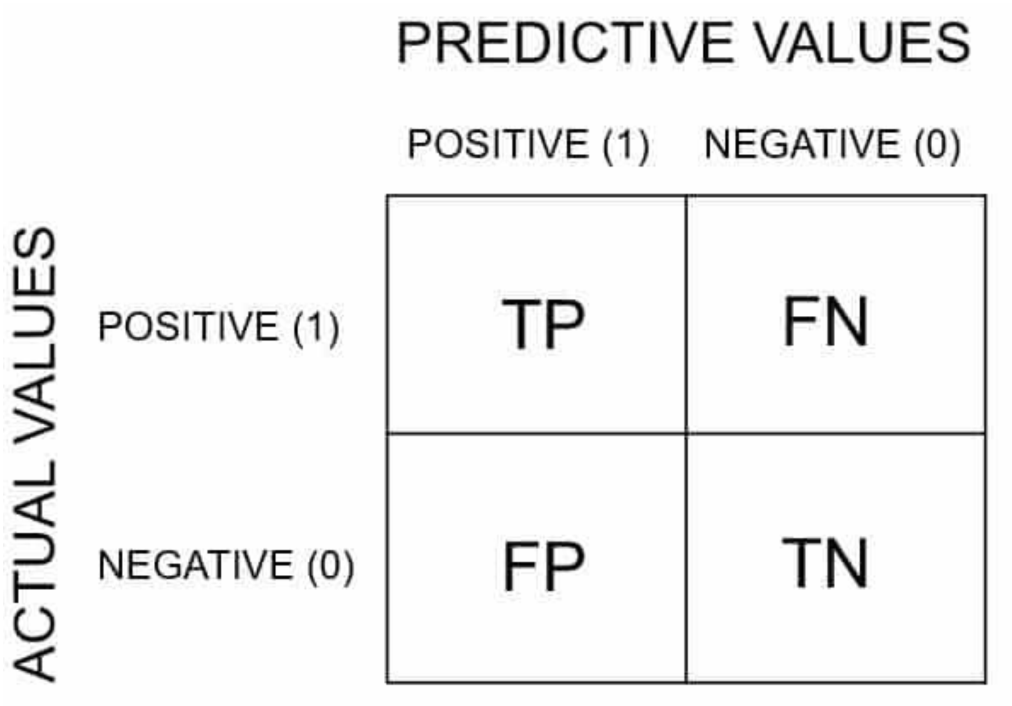
Appendix

<u>BalAcc Score</u>

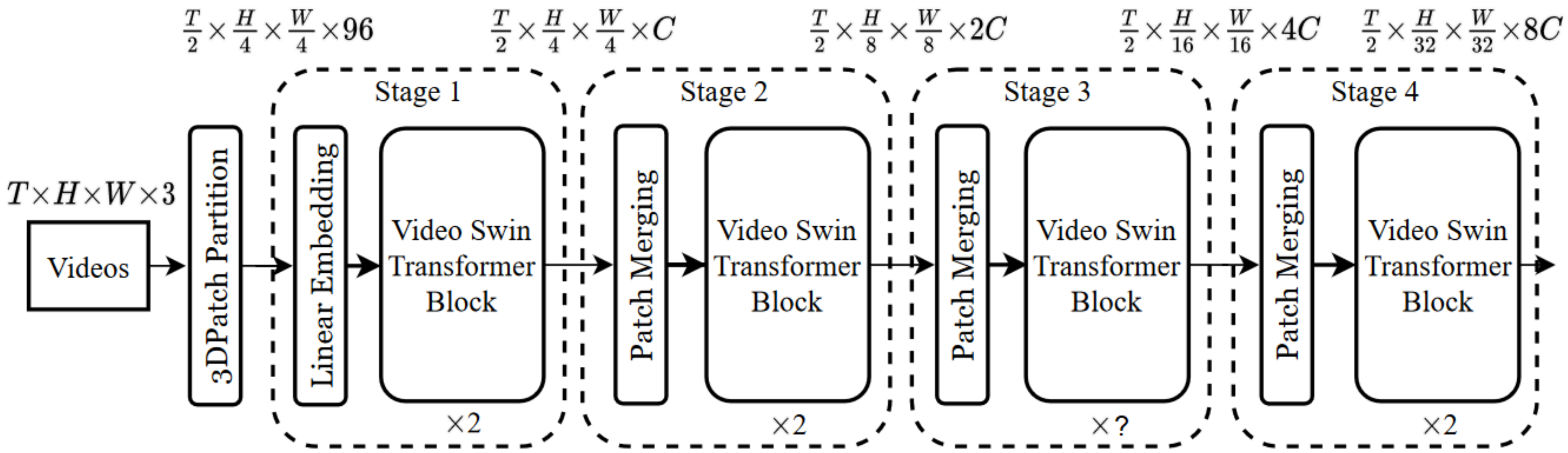
Mean class accuracy

Class imbalance issue

$$\frac{1}{N_c}\sum_{c=1}^{N_c}\frac{TP_c}{TP_c+FN_c}$$



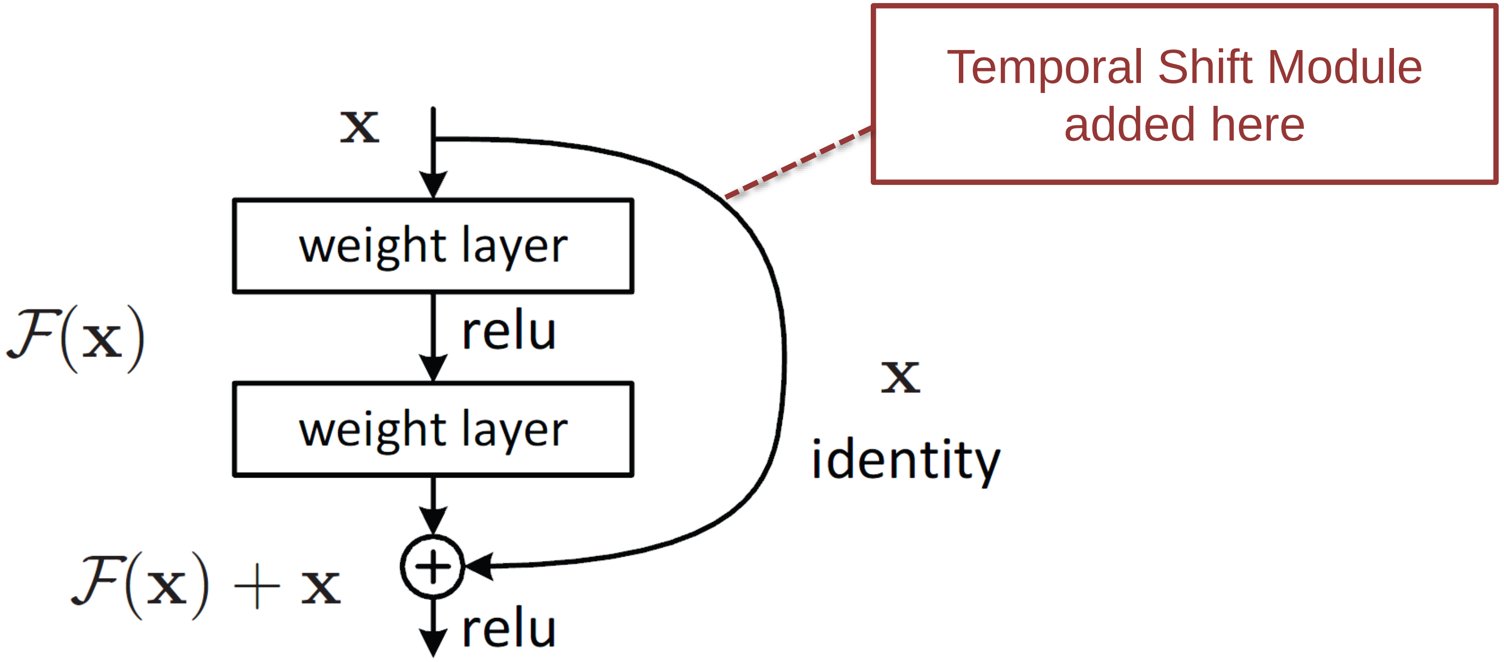
VST Architecture



- For the VST model, there are four variants officially released – tiny, small, base, and large. The variants differs in two ways: (i) the hidden channel number in the first stage (ii) the number of VST blocks in the third stage.

Z. Liu *et al.*, “Video Swin Transformer,” in *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2022, pp. 3192–3201. doi: 10.1109/CVPR52688.2022.00320.

ResNet50 + TSM Architecture



K. He, X. Zhang, S. Ren, and J. Sun, “Deep Residual Learning for Image Recognition.” arXiv, Dec. 10, 2015. doi: 10.48550/arXiv.1512.03385.

ResNet50 Architecture

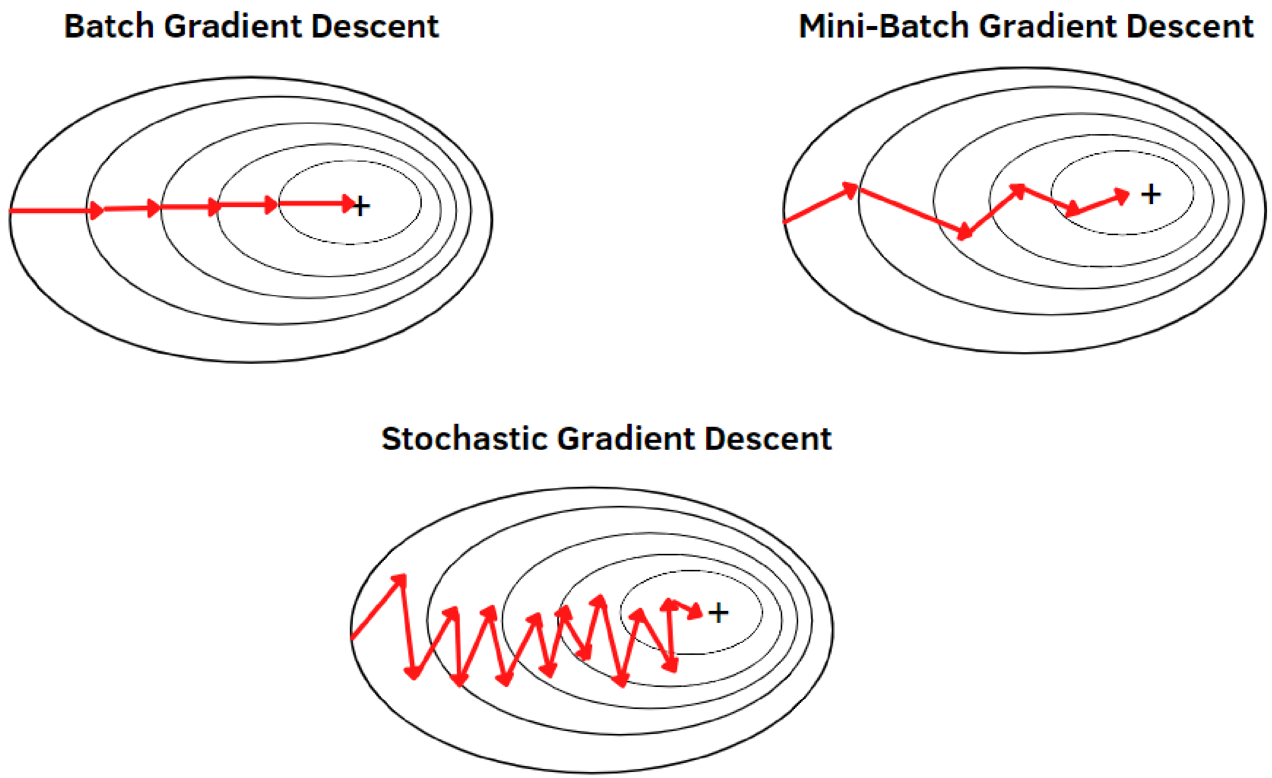
Appendix

layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
conv2_x	56×56	3×3 max pool, stride 2				
		$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$
conv3_x	28×28	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times8$
conv4_x	14×14	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times23$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times36$
conv5_x	7×7	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$
	1×1	average pool, 1000-d fc, softmax				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

K. He, X. Zhang, S. Ren, and J. Sun, “Deep Residual Learning for Image Recognition.” arXiv, Dec. 10, 2015. doi: 10.48550/arXiv.1512.03385.

Optimizer: SGD

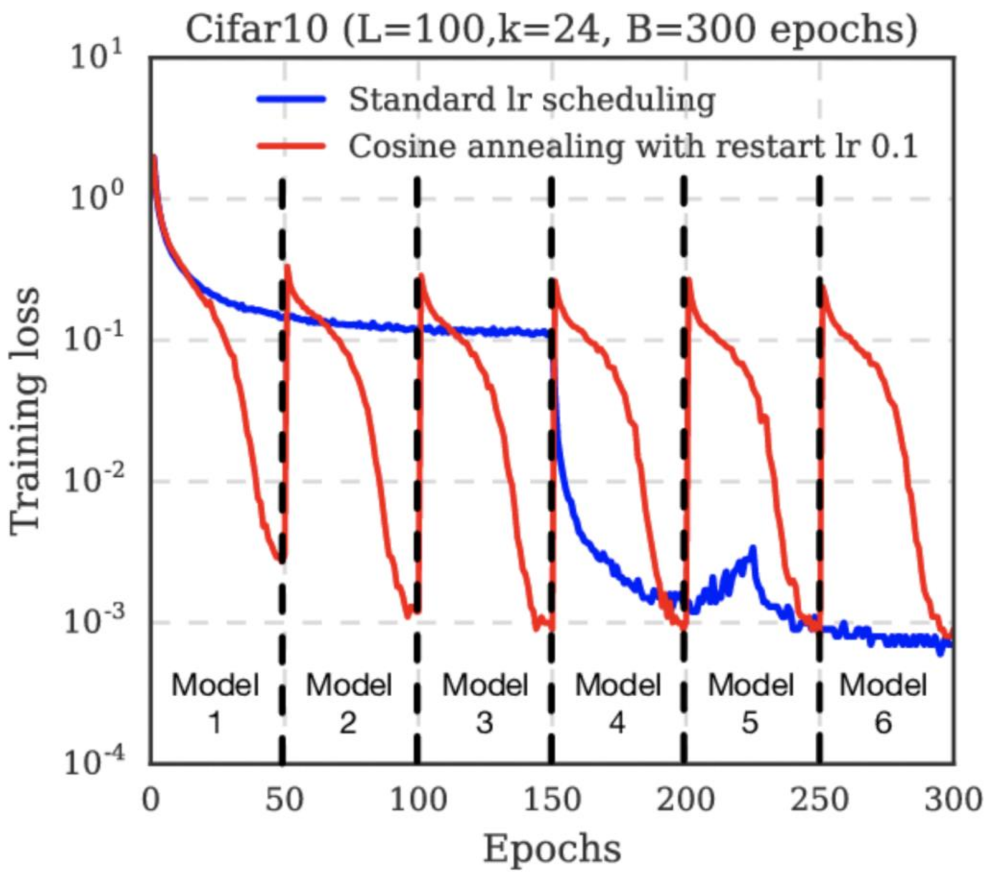
Appendix



https://www.analyticsvidhya.com/blog/2022/07/gradient-descent-and-its-types/

LR Schedulers

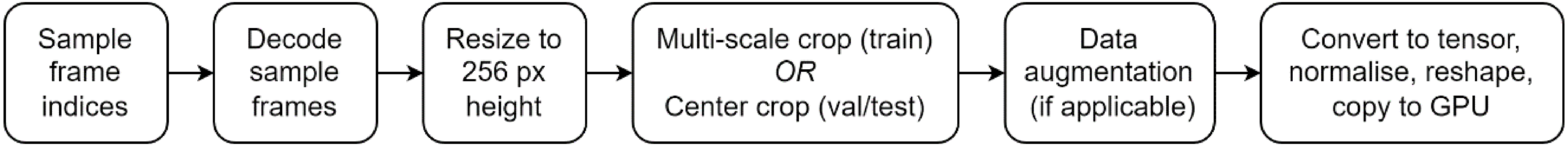
Appendix



I. Loshchilov and F. Hutter, “SGDR: Stochastic Gradient Descent with Warm Restarts.” arXiv, May 03, 2017. doi: 10.48550/arXiv.1608.03983.

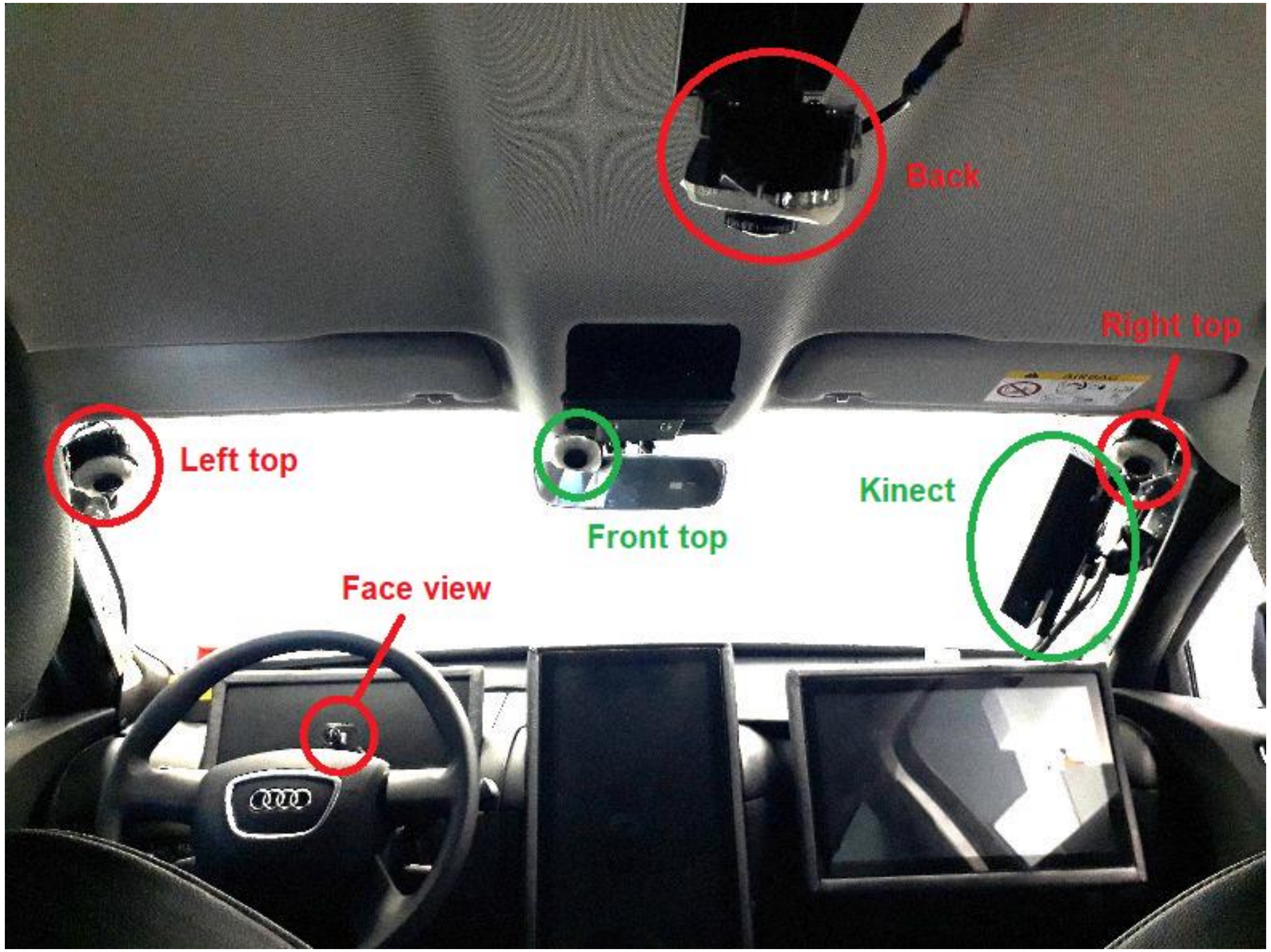
Data Transformation Pipeline

Appendix



DAA Setup

Appendix



DAA Statistics

Appendix

Table 4-1: Breakdown of sample counts for each split on DAA L2 activities.

Location	Split	Train	Val	Test	Total
Front Top	0	6642	1430	2222	10294
	1	7253	1385	1656	10294
	2	6693	1356	2245	10294
Right Top	0	6746	1459	2253	10458
	1	7374	1404	1680	10458
	2	6796	1375	2287	10458

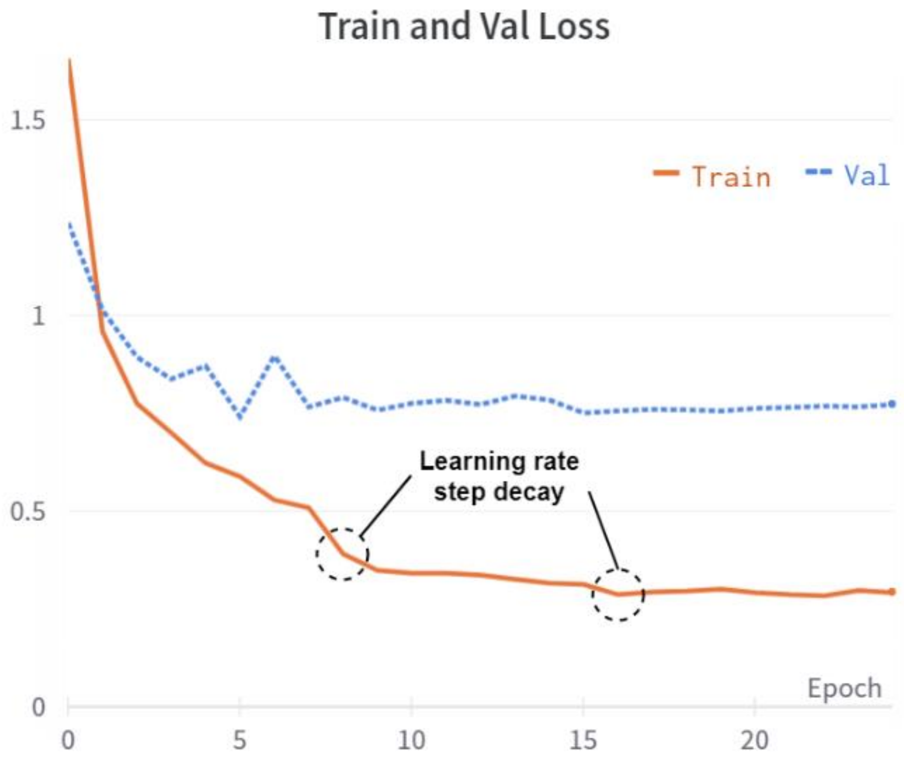
DAA Splits

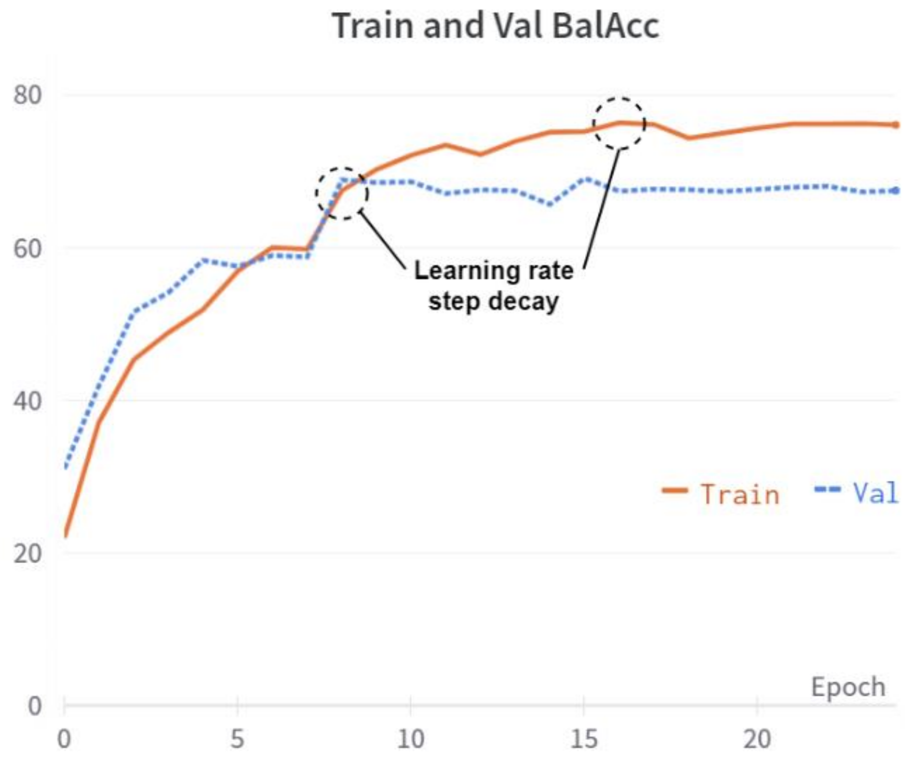
Appendix

```python
# Master split dictionary
splits_dict = {
    0: {'train': [1,2,3,4,6,7,8,9,10,12], 'val': [14,15], 'test': [5,11,13]},
    1: {'train': [1,2,4,5,6,7,11,13,14,15], 'val': [3,8], 'test': [9,10,12]},
    2: {'train': [3,5,8,9,10,11,12,13,14,15], 'val': [1,2], 'test': [4,6,7]},
}
```

Effect of Learning Rate Scheduler

Appendix





Results: TSM Single Modality

Appendix

Table 5-2: Comparison of results on DAA while varying training methods on TSM.

Model	Training Method	N_s	Val ↑	Test ↑
TSM	Baseline	8	60.17	55.24
		16	60.64	56.76
TSM	Class Weighting	8	67.31	61.90
		16	65.98	59.41
TSM	Uniform Class Sampling	8	64.33	58.13
		16	64.57	57.79
TSM	Hard Sample Mining	8	61.09	56.67
		16	62.38	55.43
TSM	CW+HSM	8	**68.47**	**62.34**
		16	66.57	59.37

Results: Single Modality

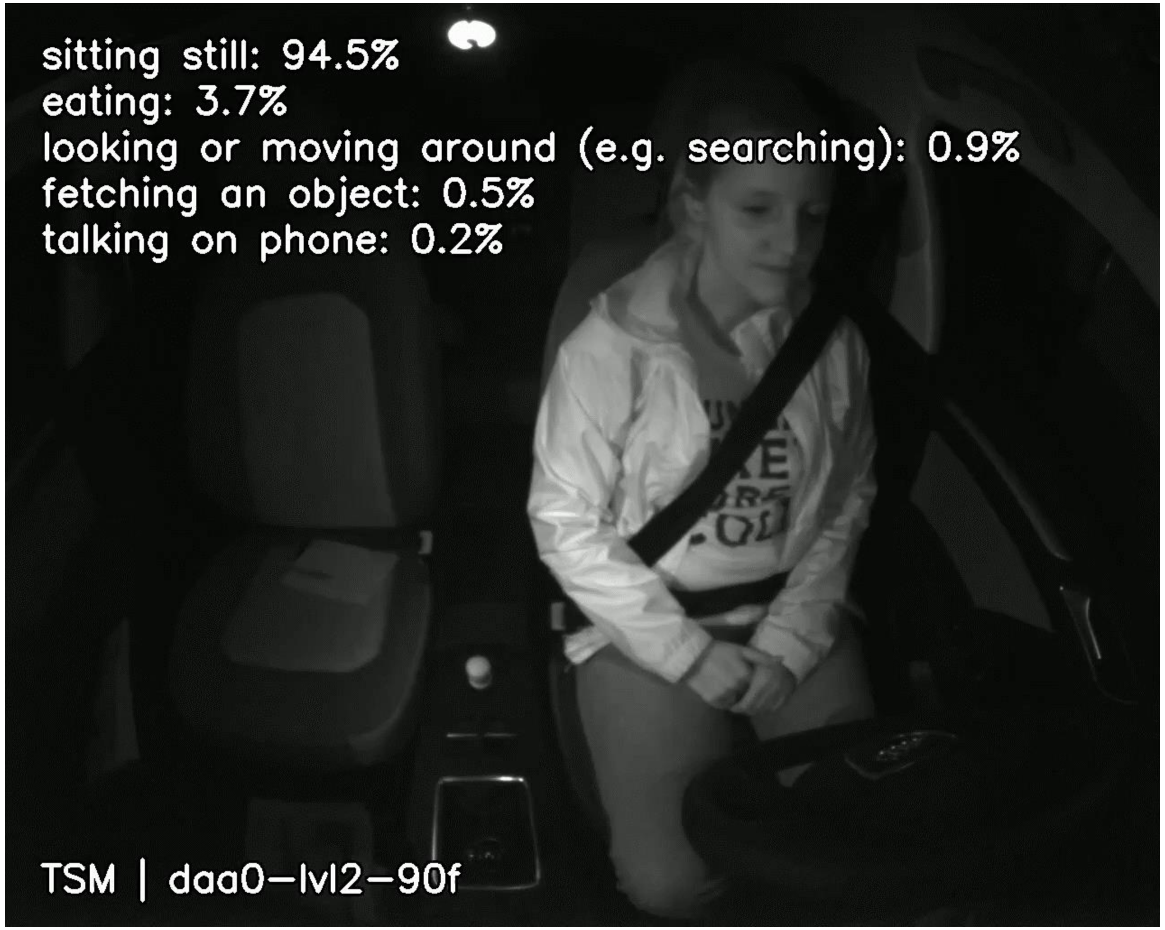
Appendix

Table 5-3: Summary of BalAcc scores on DAA between trained models.

Model	**Training Method**	N_s	**Val ↑**	**Test ↑**
$\text{I3D}_{\text{ResNet50}}$	Baseline	32	56.22	53.79
	Uniform Class Sampling		63.29	60.41
$\text{TSM}_{\text{ResNet50}}$	Baseline	8	60.17	55.24
	CW+HSM		68.47	62.34
VST_{Tiny}	Baseline	32	64.05	59.02
	Uniform Class Sampling		68.67	60.10
VST_{Base}	Baseline	32	66.44	61.89
	Uniform Class Sampling		**70.06**	**63.48**

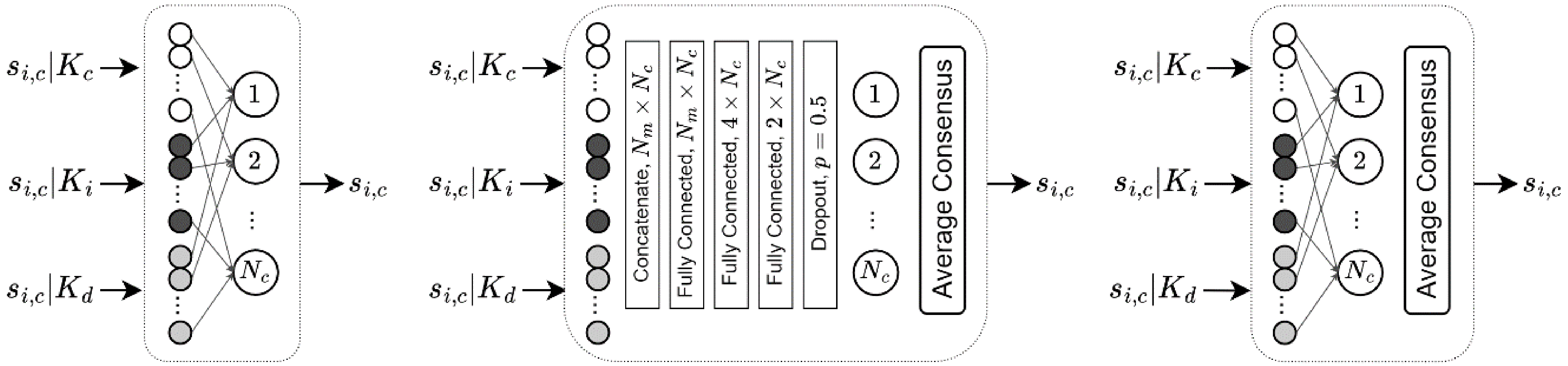
Results: Single Modality

Appendix



Multi-Modality Fusion

Appendix



(A) *Equal weights* (B) *Concatenate scores → MLP* (C) *Learned weights*

Late Fusion Type C

Appendix

```
class LinearWeightedAvg(nn.Module):

    def __init__(self, streams: List[str], n_neurons: int):
        super(LinearWeightedAvg, self).__init__()
        self.streams = streams
        self.n_neurons = n_neurons
        bound = 1 / math.sqrt(n_neurons)
        for stream in self.streams:
            setattr(self, f'weight_{stream}', nn.Parameter(torch.ones((1, n_neurons))))
            torch.nn.init.uniform_(getattr(self, f'weight_{stream}'), -bound, bound)

    def forward(self, x):
        output = torch.zeros_like(x[self.streams[0]])
        for stream in self.streams:
            output += x[stream] * getattr(self, f'weight_{stream}')
        return output
```

Results: Multi-Modality

Appendix

Table 5-5: Summary of proposed multi-modal fusion methods on DAA while varying modality combinations.

Fusion Method	Modality	Val ↑	Test ↑
No Fusion (single modality)	K_c	70.35	62.72
	K_i	69.33	59.81
	K_d	68.31	58.28
Early Fusion (concatenate channels)	$K_c + K_i$	69.12	63.46
	$K_c + K_d$	66.08	59.97
	$K_i + K_d$	69.74	60.44
	$K_c + K_i + K_d$	66.50	61.30
Late Fusion: Type A (average probabilities)	$K_c + K_i$	73.52	65.30
	$K_c + K_d$	73.25	65.19
	$K_i + K_d$	73.24	62.87
	$K_c + K_i + K_d$	73.81	64.69
Late Fusion: Type B (concatenate scores - MLP)	$K_c + K_i$	70.32	63.06
	$K_c + K_d$	66.50	61.30
	$K_i + K_d$	72.50	62.51
	$K_c + K_i + K_d$	73.42	64.60
Late Fusion: Type C (linear weighted scores)	$K_c + K_i$	71.47	64.99
	$K_c + K_d$	74.06	65.09
	$K_i + K_d$	74.02	63.24
	$K_c + K_i + K_d$	**75.20**	**66.32**

Results: Multi-Modality

Appendix

Optimized Data Transformation

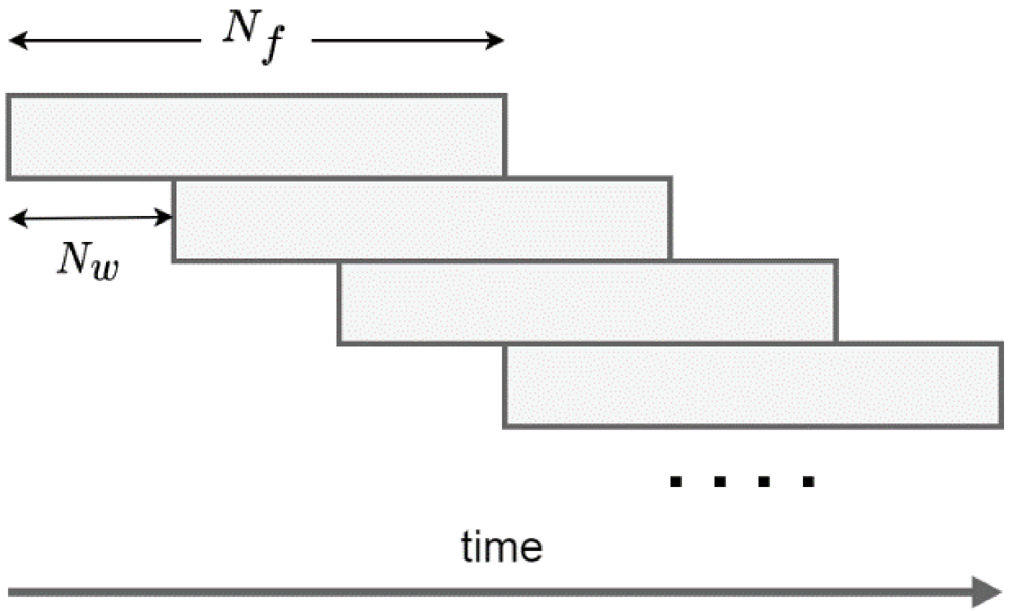
Appendix

```
def transform_fast(frames_list):
    data = [frames_list[i][:, -H:, ::-1] for i in indices]  # Right cropping
    data = [cv2.resize(img, (256, 256))[16:-16, 16:-16, :] for img in data]
    data = np.concatenate(data, axis=2)
    data = torch.from_numpy(data).permute(2, 0, 1).contiguous()
    data = data.cuda(0).float().div(255)
    data = normalize_transform(data)
    return data.unsqueeze(0)
```

Sliding Temporal Window

Appendix

- GeForce RTX 3080 GPU card
- $N_f = 90$, Number of frames
- $N_w = 30$, Step size



Live Real-time: Latency Profiles

Appendix

Table 5-7: Latency times for each stage, measured on demo workstation.

Stage	**Latency**
Load frame to buffer	0.5 ms*
Data transformation	15.6 ms
Model inference	15.8 ms
Misc. drawing and display	0.5 ms*
Total	**32.4 ms***

Full Results: Live Real-time Prototype

Appendix

Conclusion

Appendix

- Improved accuracies
  - Baseline to CW+HSM: 8 BalAcc points increase
  - Single modality to multi-modality: 5 BalAcc points increase
- 30 FPS live prototype
  - Low latency and high accuracy