A modified Single-Shot multibox Detector for beyond Real-Time Object Detection

Georgios Orfanidis†, Konstantinos Ioannidis†, Stefanos Vrochidis†, Anastasios Tefas∗ and Ioannis Kompatsiaris†

†Information Technologies Institute

Centre for Research and Technology, Hellas

∗Department of Informatics

Aristotle University of Thessaloniki

CENTRE FOR RESEARCH & TECHNOLOGY - HELLAS
Information Technologies Institute

ARISTOTLE UNIVERSITY OF THESSALONIKI

Outline

- INTRODUCTION
- RELATED WORK
- METHOD
  - Adjusted loss classification weights
  - Selecting the proper decision layers
- EXPERIMENTS
  - Balancing dataset
  - Results on Pascal Voc 2007 dataset
  - Results on KITTI dataset
- CONCLUSION

INTRODUCTION

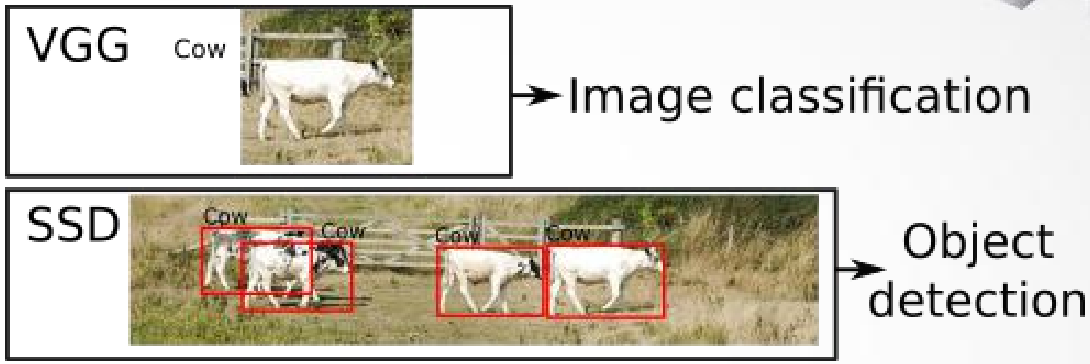
- ❑ Object detection remains a fundamental problem in computer vision
- ❑ Objective: **localize** (provide a bounding box) and **identify** (provide a label) for objects of interest inside an image.

- ❑ Solution: **Convolutional Neural Networks (CNN)** lead to huge improvements
  - ➢ Typical State-of-the-Art models are **computationally expensive**
  - ➢ Restricted integration on systems with limited resources.
  - ➢ **Lighter versions** have emerged: Tiny-YOLO, SqueezeDet, MobileNet-SSD

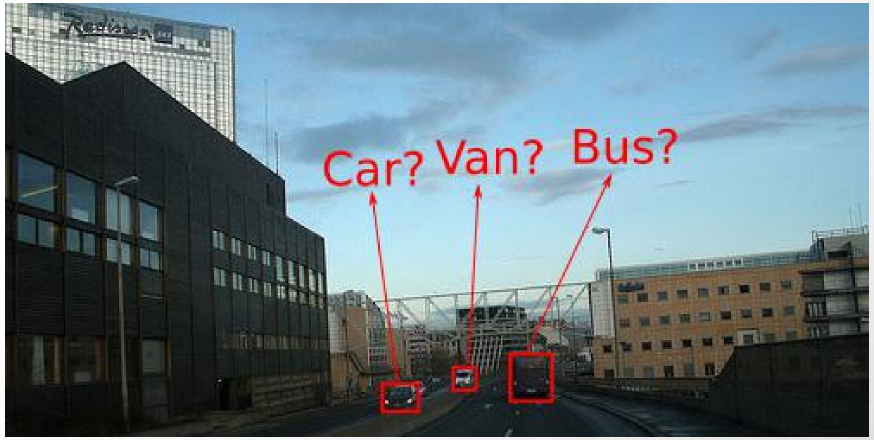
RELATED WORK

- Object detection is divided into two major categories based on the potential use of a **Region Proposal Network (RPN)**:
  - the single-phase detectors
    - SSD, YOLO, YOLOv2, Retinanet etc
  - the two-phase detectors
    - Fast R-CNN, Faster R-CNN and R-FCN
- Another categorization regarding the **object detection models' purpose**:
  - **state-of-the-art performances** with no resource restrictions
  - best performance in **resource restricted environments**
  - It is almost exclusively dominated by the **single-phase detectors** due to the efficiency they inherently possess

METHOD 1/3



- ❑ Original SSD modifies VGG network.
- VGG is a robust network but:
  - ➢ Uses huge number of parameters, nonetheless
  - ➢ limited use in resource-restricted applications.
- ❑ SSD suffers in identifying small objects.
  - ➢ The shallowest layer which is being used is **conv4_3** of VGG
  - ➢ typical input size 300x300 → corresponds to a **38x38 feature map**
  - ➢ too small to identify objects
- ❑ SSD includes 10 blocks of CNNs in order to extract features.
  - ➢ first **6 blocks** belong to the VGG
  - ➢ each next block has **double the filters** of the previous one
  - ➢ The initial number of filters is 64 for the 1st block.



METHOD 2/3

- We added an extra shallower decision layer at **conv3_3**
  - with **75x75 feature map**
  - number of default boxes number 8732 → 31232
  - Are shallower features discrimant enough?
- Decreased both the initial number of filters as well as the exponent for increase for the next blocks.
- $k_n = b^{an}$
  - Initial numbers of filters, **parameter b**, 48 and 32 were examined
  - **parameter a** was fixed to 1.7 (from 2 to the original VGG)

METHOD 3/3

- **Number of filters** used in the various adaptations

	block name	Formula for #filters		
		full SSD 64^2	SSD_lite_48 $48^{1.7}$	SSD_lite_32 $32^{1.7}$
VGG layers	$conv1\_x$	64	48	32
	$conv2\_x$	128	81	54
	$conv3\_x$	256	138	92
	$conv4\_x$	512	235	157
	$conv5\_x$	512	235	157
	fc_x	1024	400	267
Additional layers	$conv6\_x$	256/512	138/235	92/157
	$conv7\_x$	128/256	81/138	54/92
	$conv8\_x$	128/256	81/138	54/92
	$conv9\_x$	128/256	81/138	54/92

Adjusted loss classification weights

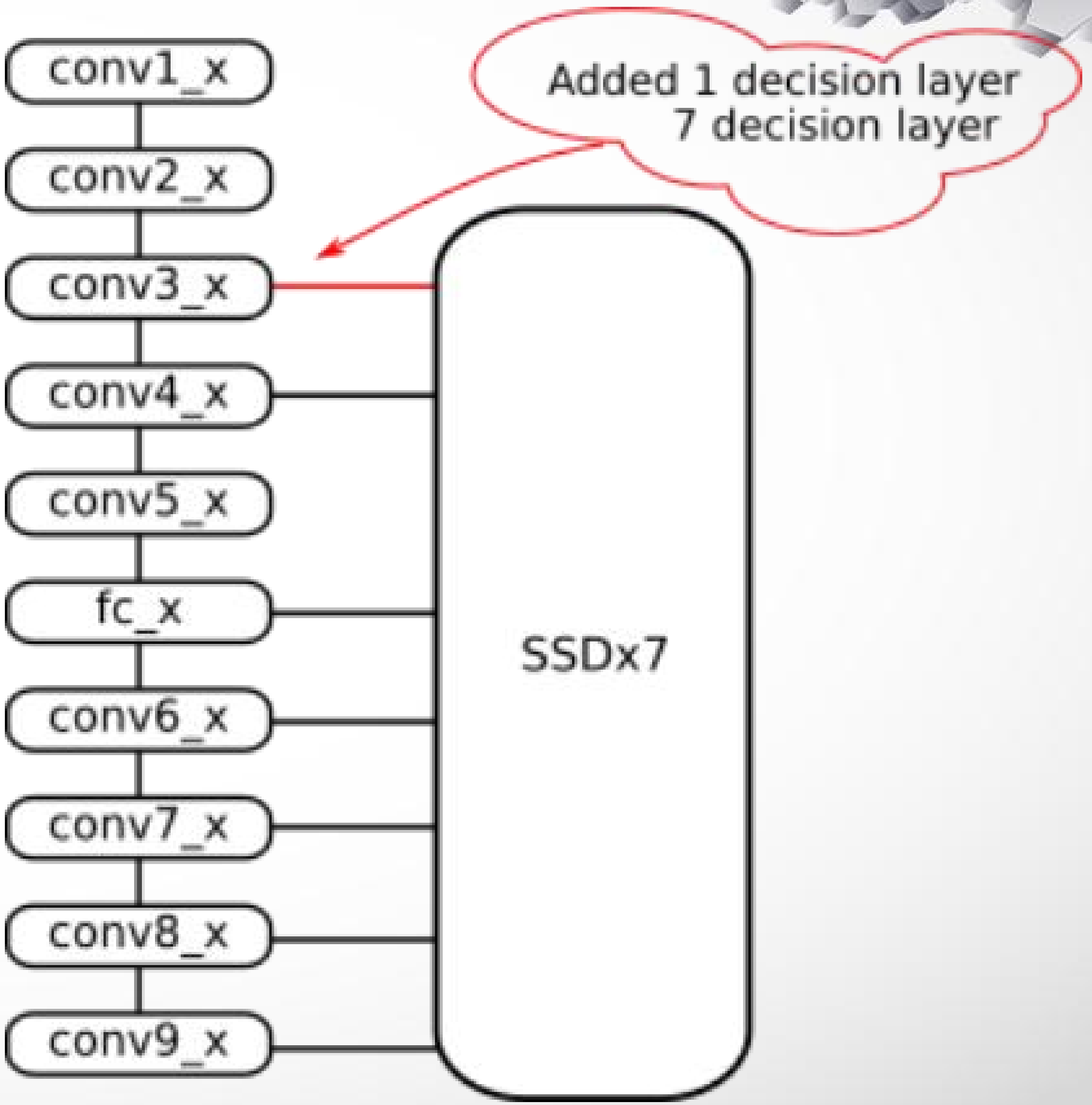
- Compensate for unbalanced datasets
- **Modified version** of SSD classification **loss function**
  - different weight coefficients for different classes
- **KITTI** dataset:
  - $loss = w_{ped} * loss_{ped} + w_{cycl} * loss_{cycl} + w_{car} * loss_{car}$
  - $w_{ped} = 2.2$, $w_{cycl} = 2.0$, $w_{car} = 1.0$
- **Pascal Voc** dataset:
  - $loss = w_1 * loss_1 + \ldots + w_{20} * loss_{20}$
  - $w_i = \frac{AP_{cat}}{AP_i}$

Cat class has the best performance
(used as reference class)

- Improves performance for classes of **lower overall performance**

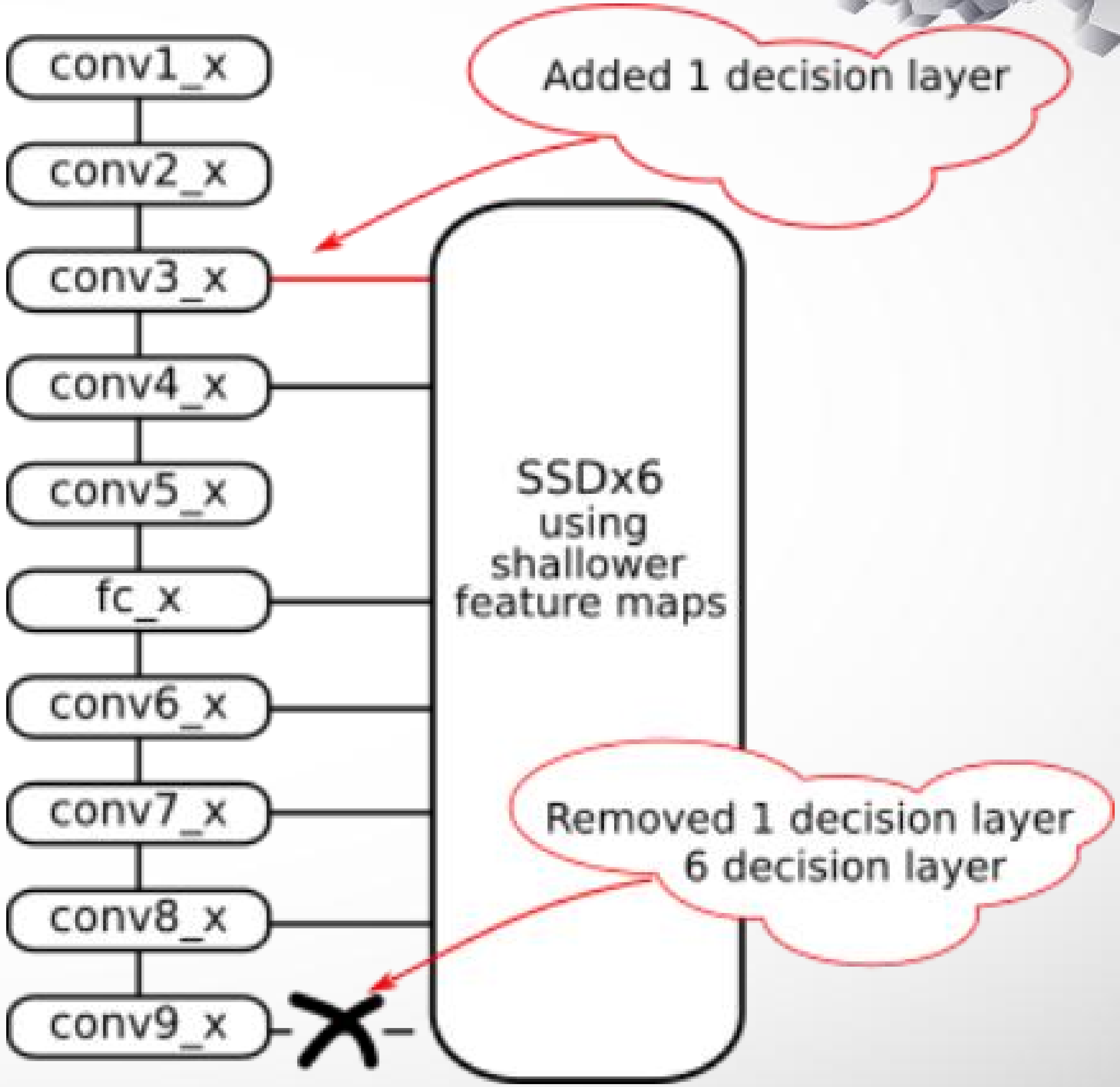
Selecting the proper decision layers 1/5

- SSD deployed 6 decision layers
  - They are used to **extract discriminant features.**
  - Each one with different feature map size.
- Formation of **SSDx7**
  - **1 additional shallower decision layer**
  - Better performance in KITTI
  - Decreased performance In Pascal Voc



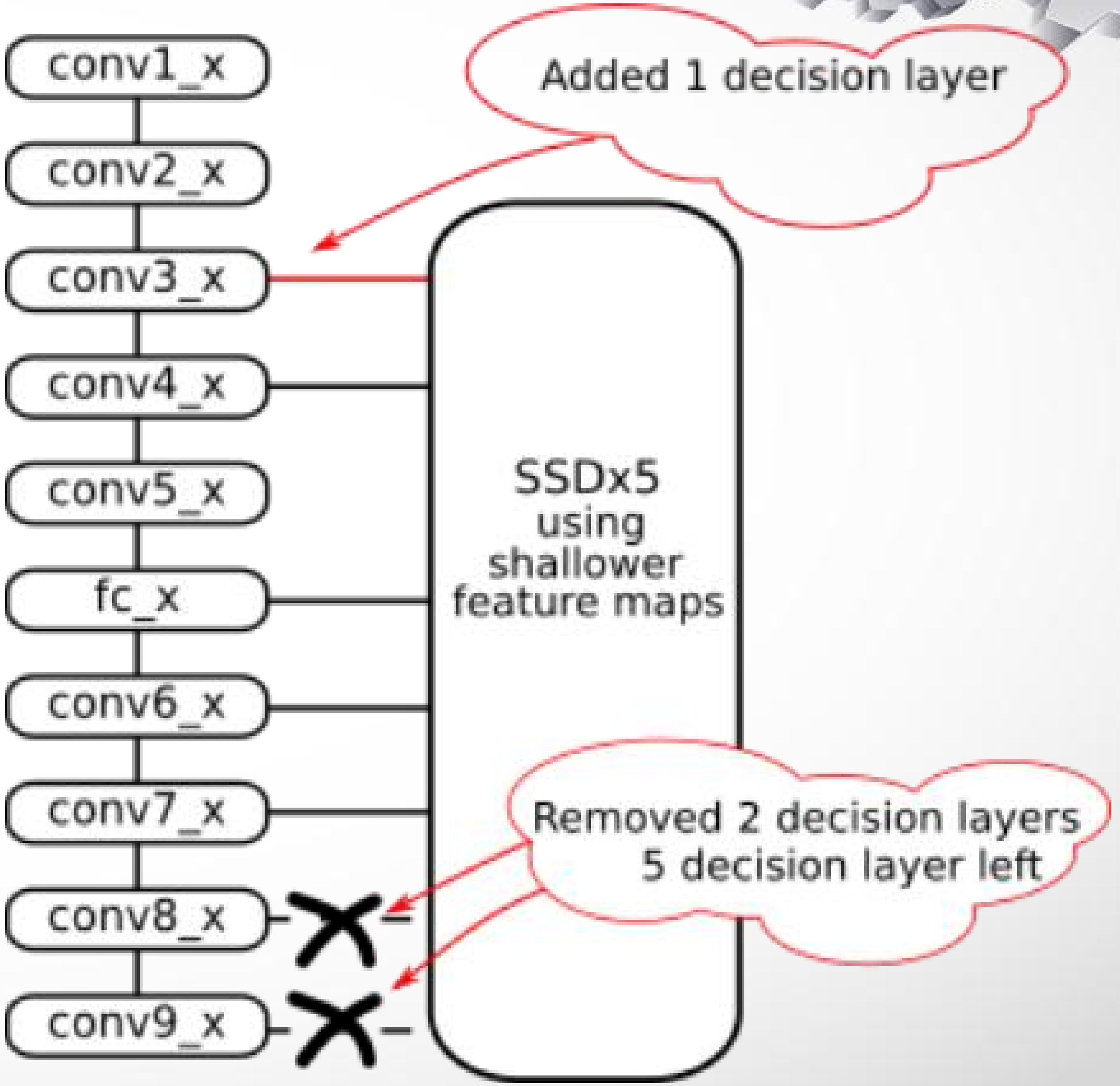
Selecting the proper decision layers 2/5

- ❑ Formation of **shallower SSDx6**
  - ❑ **1 additional shallower decision layer** used.
  - ❑ **1 deeper layer** being **removed**
  - ❑ The (removed) **deepest layer** useful for bigger objects only.
    - ➢ They do not appear in KITTI
    - ➢ Are non frequent in Pascal



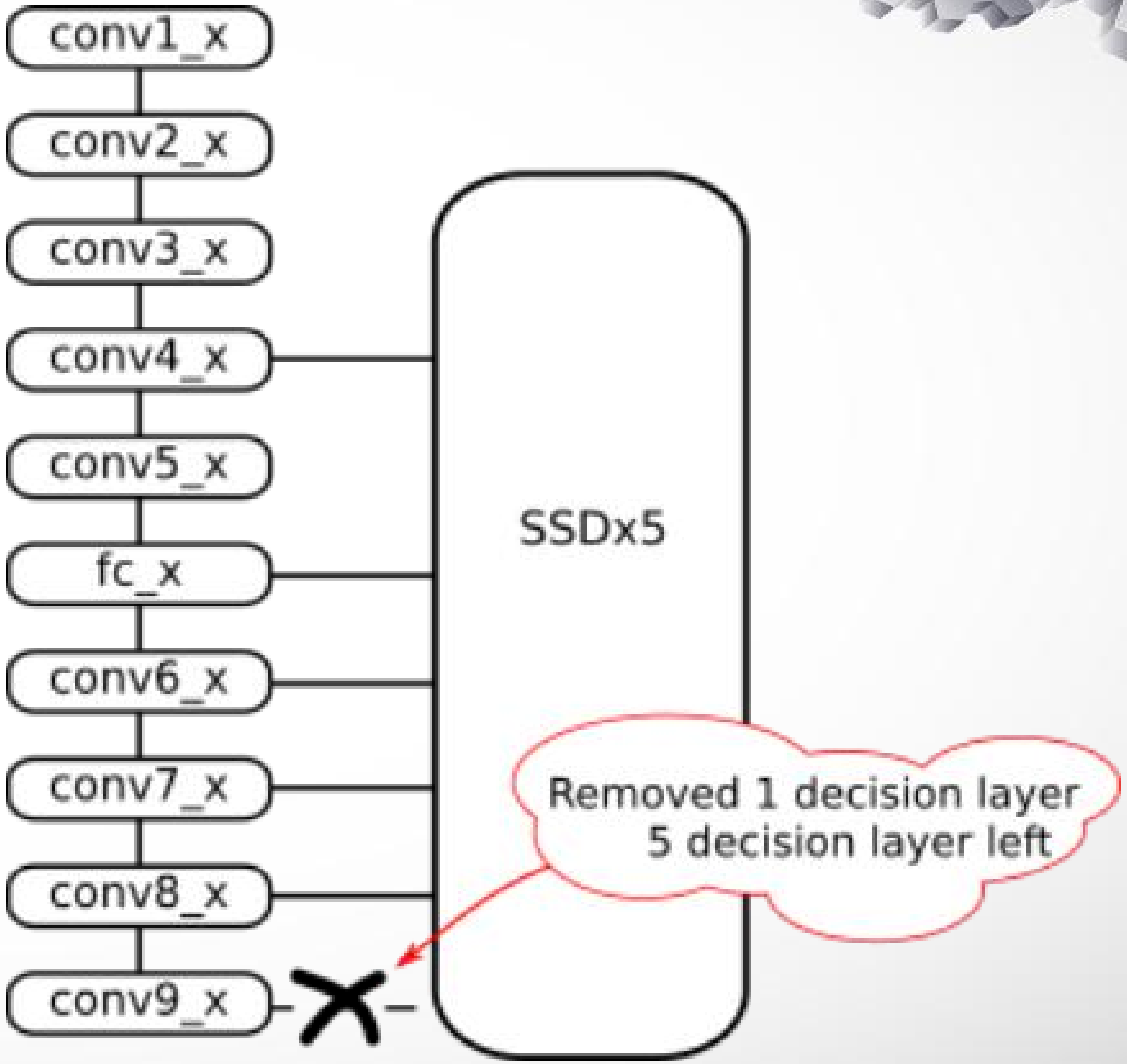
Selecting the proper decision layers 3/5

- Formation of **shallower SSDx5**
  - **1 additional shallower decision layer**
  - 2 **deeper layers** were removed
  - Only well performing in KITTI



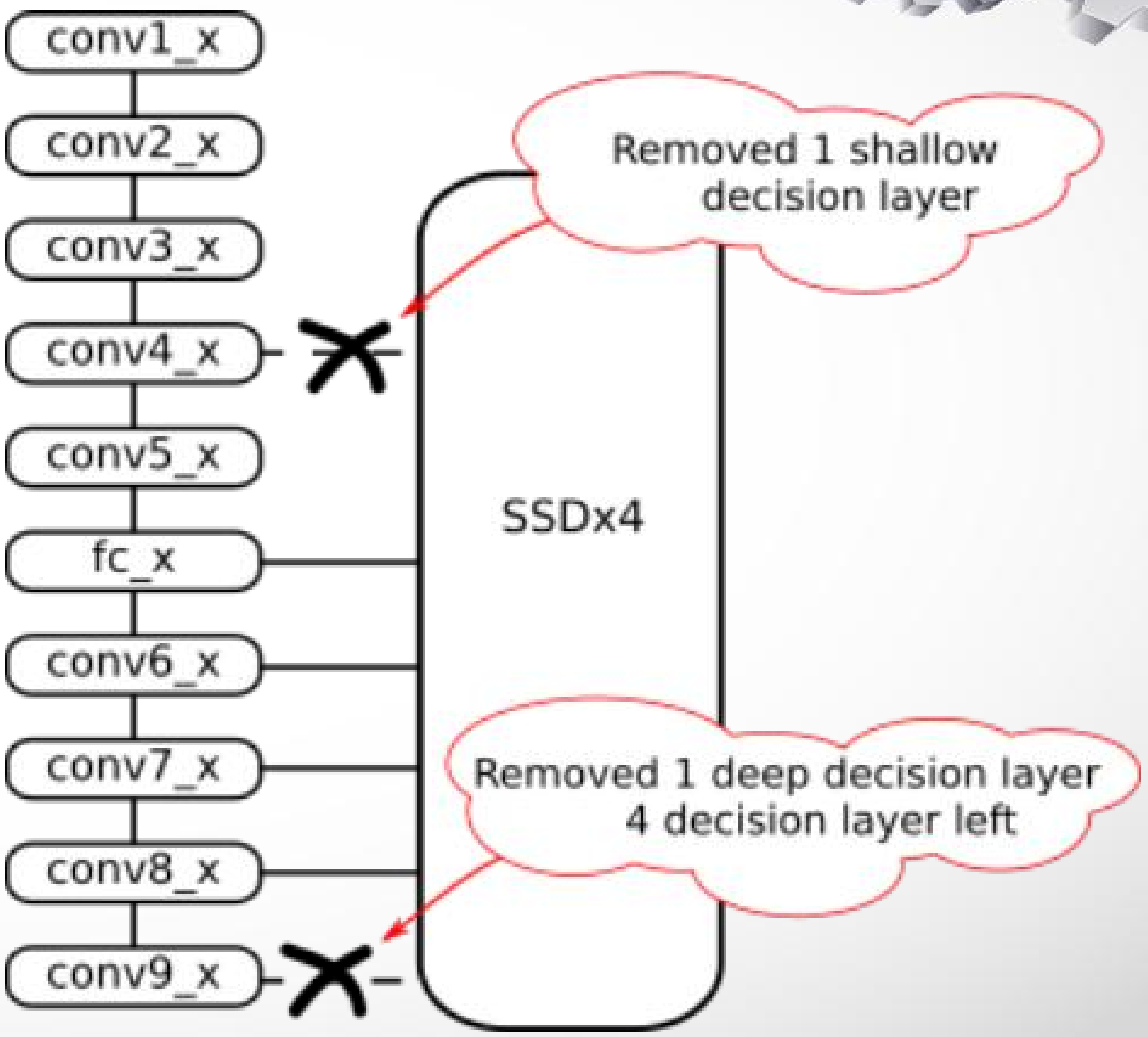
Selecting the proper decision layers 4/5

- Formstion of **SSDx5**
  - 1 **deeper layer** was removed
  - Only well performing in Pascal Voc



Selecting the proper decision layers 5/5

- Formation of **SSDx4**
  - 1 **deeper layer** was removed
  - 1 **shallower layer** was also removed
  - Only performing well in Pascal Voc



EXPERIMENTS - Balancing the dataset

- **Experiments** were conducted in Pascal Voc and KITTI datasets
- Both datasets are imbalanced.
- Repeat images containing objects from misperforming classes
- Useful for KITTI not for Pascal
- Might **improve** the performance of **some classes** but **decrease** the performance for the remaining classes.

This image could be repeated in the training dataset

Results on Pascal Voc 2007 dataset 1/4

- Full model adaptations:
  - Incorporating an additional shallower layer **did not increase** the performance**.**
  - **Weighted** version of **SSDx6**, **SSDx5** and **SSDx6** all tie at 77.6%
  - Performance of **3 worst classes** did **improve on Weighted version**.

Model name	Decision Layers (train/eval)			mAP
	num	initial	last	
Full SSDx6	6/6	*conv4_3*	*conv9_2*	**77.6%**
Full SSDx6 vs5	6/5	*fc7*	*conv9_2*	71.2%
Full SSDx6 vs5	6/5	*conv4_3*	*conv8_2*	**77.6%**
Full SSDx7	7/7	*conv3_3*	*conv9_2*	77.5%
w. Full SSDx6	6/6	*conv4_3*	*conv9_2*	**77.6%**

Removing last layer

Using an extra layer

Weighted full version

Model name	Average Precision (AP)			
	bottle	chair	potted plant	3 class
Full SSDx6	50.50%	60.90%	53.60%	53.90%
Full SSDx7	49.20%	59.20%	53.30%	55.00%
w. Full SSDx6	**52.50%**	**61.50%**	**54.50%**	56.17%

3 worst performing classes

Results on Pascal Voc 2007 dataset 2/4

- **Medium model** adaptations:
  - **Removing** shallower layer **did not improve the overall performance** (almost 5% compared to baseline)**.**
  - Inclusion of **Last layer did not affect the results.**
  - **Weighted** version of **SSDx4** model demonstrated best performance at 71.0% mAP.

SSD lite 48x6	6/6	*conv4_3*	*conv9_2*	61.7%
SSD lite 48x6 vs5	6/5	*fc7*	*conv9_2*	66.6%
SSD lite 48x6 vs5	6/5	*conv4_3*	*conv8_2*	61.6%
SSD lite 48x4	4/4	*fc7*	*conv9_2*	70.6%
w. SSD lite 48x4	4/4	*fc7*	*conv9_2*	**71.0%**

Full medium model
baseline for medium model

Removed shallower layer

Weighted truncated version

Results on Pascal Voc 2007 dataset 3/4

- **Lighter model** adaptations:
  - **Removing** shallower layer **improved performance** (4% compared to baseline).
  - **Last layer do not affect results.**
  - **Weighted** version **SSDx4** model demonstrated best performance at 64.1% mAP

SSD lite 32x6	6/6	*conv4_3*	*conv9_2*	55.9%
SSD lite 32x6 vs5	6/5	*conv4_3*	*conv8_2*	55.9%
SSD lite 32x6 vs5	6/5	*fc7*	*conv9_2*	59.9%
SSD lite 32x4	4/4	*fc7*	*conv8_2*	63.1%
w. SSD lite 32x4	4/4	*fc7*	*conv8_2*	**64.1%**

Full light model
baseline for light model

Removed shallower layer

Weighted truncated version

Results on Pascal Voc 2007 dataset 4/4

- Various **light-weight models' performance** on **Pascal Voc 2007 test set**:

Model name	Num Decision Layers	mAP
Tiny-DSOD	6	72.1%
w. SSD lite 48x4	4	71.0%
Pelee	4	70.9%
MobileNet-SSD	4	68.1%
w. SSD lite 32x4	4	64.1%

Results on KITTI dataset 1/4

- **Full model** adaptations:
  - **A balanced dataset** was used.
  - Additional **shallower layer improved the performance** significantly.
  - **Shallower SSDx5** was used.
  - **Weighted** version of **shallower SSDx5** demonstrated best performance with mAP 86.1%.

Model name	Decision Layers (train/eval)			mAP
	num	initial	last	
Full SSDx5 b[1.5,1.5]	5/5	*conv3_3*	*conv7_2*	85.4%
w. Full SSDx5 b[1.5,1.5]	5/5	*conv4_3*	*conv7_2*	**86.1%**

Removed last 2 layers
Added 1 shallower layer

Weighted shallower version

Using balanced dataset with
proportion 2.5(=1.5+1) to 1

Results on KITTI dataset 2/4

- **Medium model** adaptations:
  - **Balancing the dataset** improved to a point (best choice additional 1.5x of the original samples).
  - Additional **shallower layer improved performance** significantly (50%+).
  - **Weighted** version of **shallower SSDx5** demonstrated best performance at 84.1% mAP.

Unbalanced (original) dataset

SSD lite 48x6	6/6	*conv4_3*	*conv9_2*	23.2%
SSD lite 48x7	7/7	*conv3_3*	*conv9_2*	75.0%
SSD lite 48x7 b[1,1]	7/7	*conv3_3*	*conv9_2*	81.1%
SSD lite 48x7 b[1.5,1.5]	7/7	*conv3_3*	*conv9_2*	81.6%
SSD lite 48x7 b[1.5,1.5]	7/6	*conv3_3*	*conv8_2*	81.6%
SSD lite 48x7 b[1.5,1.5]	7/5	*conv3_3*	*conv7_2*	81.6%
SSD lite 48x7 b[2,2]	7/7	*conv3_3*	*conv9_2*	80.8%
SSD lite 48x5 b[1.5,1.5]	5/5	*conv3_3*	*conv7_2*	82.0%
w. SSD lite 48x5 b[1.5,1.5]	5/5	*conv3_3*	*conv7_2*	**84.0%**

Original layers underperformed

Balancing dataset helps to a point

Weighted shallower version

Results on KITTI dataset 3/4

- ❑ **Lighter model** adaptations:
  - ❑ Using **a balanced dataset**.
  - ❑ **Weighted** version of **shallower SSDx5** demonstrated best performance at 81.1% mAP.

SSD lite 32x7 b[1.5,1.5]	7/7	*conv3_3*	*conv9_2*	77.4%
SSD lite 32x5 b[1.5,1.5]	5/5	*conv3_3*	*conv7_2*	79.2%
w. SSD lite 32x5 b[1.5,1.5]	5/5	*conv3_3*	*conv7_2*	**81.1%**

Weighted shallower version

Using balanced dataset with proportion 2.5(=1.5+1) to 1

Results on KITTI dataset 4/4

- **Lightweight model performance** on **KITTI**:
  - Our medium model (**SSDx5)** demonstrated best performance.

Model name	Num Decision Layers	mAP
w. SSD lite 48x5 b[1.5, 1.5]	5	84.0%
w. SSD lite 32x5 b[1.5, 1.5]	5	81.1%
SqueezeDet+	1	80.4%
Tiny-DSOD	6	77.0%

Efficiency results

- **Efficiency** comparison with other **lightweight models**:
  - **Reported times** are **indicative** due to hardware differences

Model name	Resolution	batch size	fps	GPU
Full SSDx6	300x300	1	44	GTX 1070 Ti 8GB
SSD lite 48x4	300x300	1	59	GTX 1070 Ti 8GB
SSD lite 32x4	300x300	1	90	GTX 1070 Ti 8GB
Pelee	304x304	1	77	TX2 (32FP)*
Tiny-DSOD	300x300	1	105	TitanX
MobileNet-SSD	300x300	1	59.3	TitanX
Full SSDx5	620x300	1	29	GTX 1070 Ti 8GB
SSD lite 48x5	620x300	1	51	GTX 1070 Ti 8GB
SSD lite 32x5	620x300	1	61	GTX 1070 Ti 8GB
SqueezeDet+	1242x375	1	32.1	TitanX
Tiny-DSOD	1200x300	1	64.9	TitanX
* excluding post processing time				

CONCLUSION

- **Light-weight** versions of the SSD architecture were examined.
- **Two** widely used **datasets** were utilized: Pascal Voc & KITTI.
- SSD remains **competitive** even when **many** of the original **filters were removed.**
- **Decision layer selection** affected **significantly** the **performance** especially on lighter versions.
- Effectiveness drop counter-measures proved useful:
  - **Class weights manipulation** played an important role.
  - **A balanced dataset** also improved performance (only in KITTI).

CONCLUSION

- ❑ **Thank you.**
- ❑ **Any questions?**

This work was supported by ROBORDER and ARESIBO projects funded by the European Commission under grant agreements No 740593 and No 833805 respectively.

ROBORDER Autonomous Swarm of Heterogeneous Robots for Border Surveillance

ARESIBO