Digital Rail Summer School 2022

Aspects of robustness in ML/AI object detection models

Intel Deutschland GmbH

Ralf Gräfe, ralf.graefe@intel.com; Michael Paulitsch, michael.Paulitsch@intel.com

intel®

Key Points

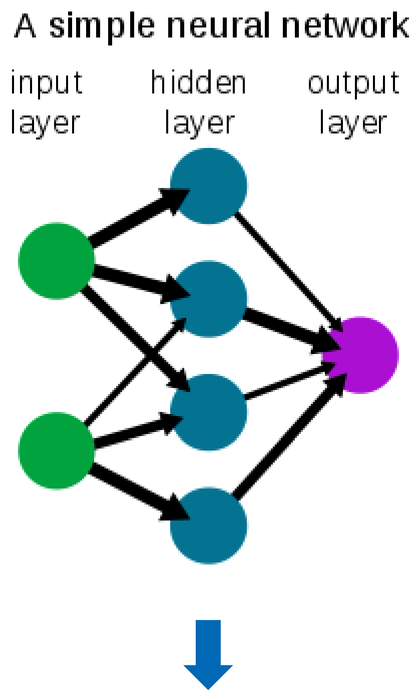
Agenda

- general introduction to NN and their application for safety critical use cases
- potential problems in the application of NN
- options to improve the safety of NN

Neural Network basics

Basic principles of NN

A neural network is a series of functions that attempt to recognize underlying relationships in a set of data



$$f = ax_1 + bx_2 + cx_3 + \ldots + zx_n$$

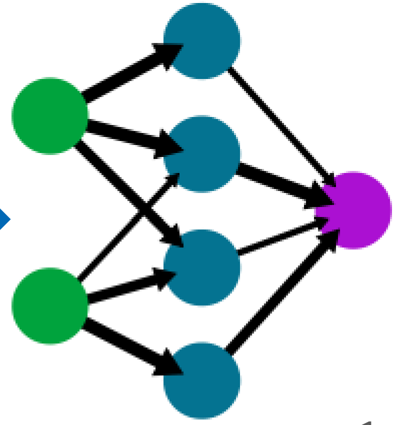
where a … z are the weights of the network and x1 … xn are the neurons

=> the goal is to optimize the weights

Basic principles of NN – training a network

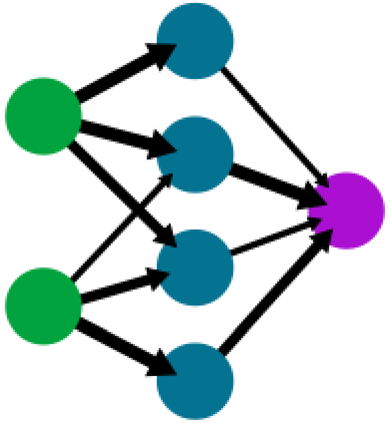
1

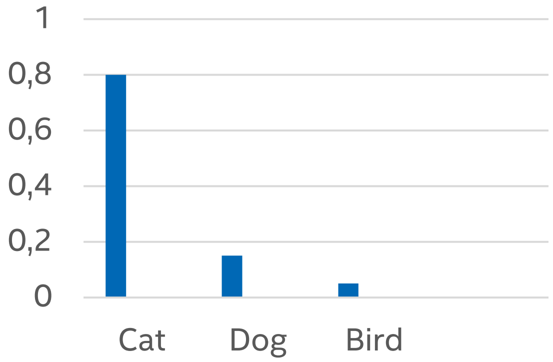
feed data into the network
that has meaning for us



2

network produces output
we can measure





3

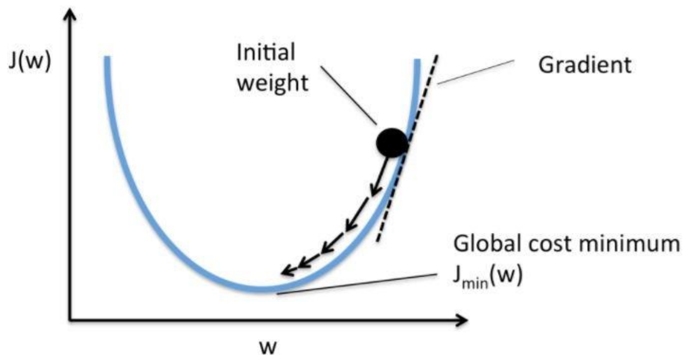
measure the distance
between true meaning and
output

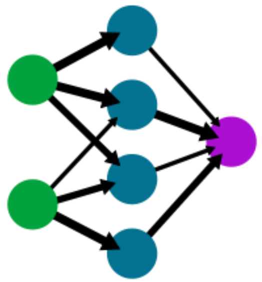
loss function = true meaning – output

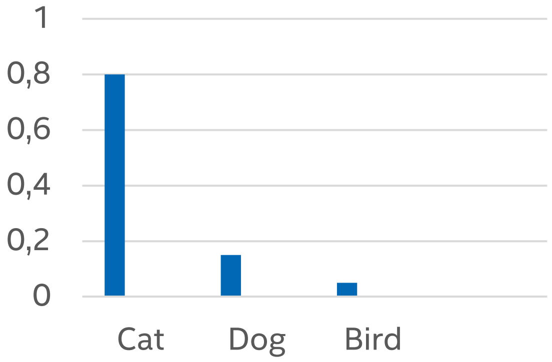
Basic principles of NN – training a network

4

Us the power of calculus to calculate gradients to reduce the difference between true meaning and output





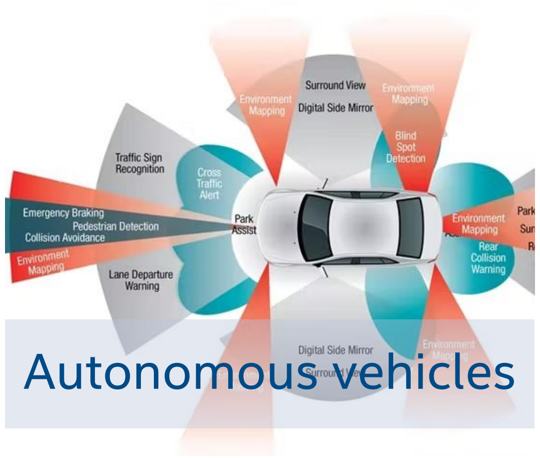


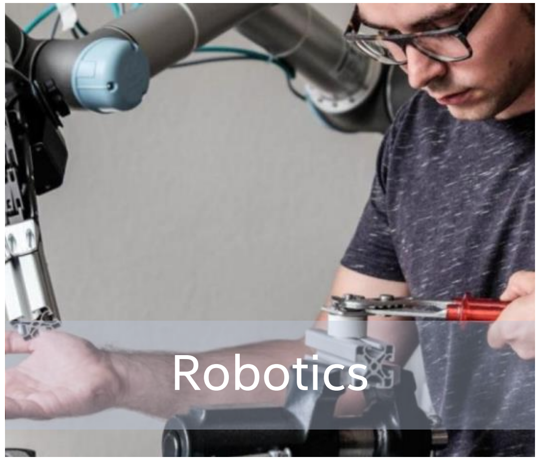
5

start over

Examples of safety critical application of NN

- detect drivable path
- detect surrounding objects
- predict trajectories of other vehicles





- detect human co-workers
- AMR obstacle detection

- Fault diagnosis
- high performance auto piloting
- Pilot supervision (detection of unsafe actions or distraction)
- Air traffic management
- UAV (Unmanned Air Vehicle) object avoidance

- Obstacle detection
- Detect persons at railroad crossing
- detect people around street cars
- Diagnosis tasks (defects of rail infras
- Checkpoints (support of defect detection like hot brakes, loose cargo, ….

Image sources:
GE; machinedesign; calaero; singaporeair

Example use cases for rail vehicles

Example use of ML

- Rolling stock (wagon) monitoring (more efficient checkpoints using cameras)
  - Like hot brakes detection, loose cargo, …

Checkpoint for supervision of rolling stock

- Railroad crossing observation (early abnormality detection like occupied tracks – especially if no bars present)
- Optimizing schedule & anticipation of irregularities

Railroad crossing monitoring possible

Main Rail / Subway Automation
Grade of Automation – Functions Needed

- Object detection on track (abnormal situations)
- Passenger observation
- Emergency situation detection
- …

With Increasing GOA

Basic functions of train operation		On-sight	Non-Automated	Semi-Automated	Driverless	Unattended
		GOA0	GOA1	GOA2	GOA3	GOA4
Ensure safe movement of trains	Ensure safe route	Ops Staff (route by systems)	Systems	Systems	Systems	Systems
	Ensure safe separation of trains	Ops Staff	Systems	Systems	Systems	Systems
	Ensure safe speed	Ops Staff	Ops Staff (partial by system)	Systems	Systems	Systems
Drive train	Control acceleration and braking	Ops Staff	Ops Staff	Systems	Systems	Systems
Supervise guideway	Prevent collision with obstacles	Ops Staff	Ops Staff	Ops Staff	Systems	Systems
	Prevent collision with persons on tracks	Ops Staff	Ops Staff	Ops Staff	Systems	Systems
Supervise passenger transfer	Control passengers doors	Ops Staff	Ops Staff	Ops Staff	Ops Staff	Systems
	Prevent injuries to persons between cars or between platform and train	Ops Staff	Ops Staff	Ops Staff	Ops Staff	Systems
	Ensure safe starting conditions	Ops Staff	Ops Staff	Ops Staff	Ops Staff	Systems
Operate a train	Put in or take out of operation	Ops Staff	Ops Staff	Ops Staff	Ops Staff	Systems
	Supervise the status of the train	Ops Staff	Ops Staff	Ops Staff	Ops Staff	Systems
Ensure detection and management of emergency situations	Detect fire/smoke and detect derailment, detect loss of train integrity, manage passenger requests (call/evacuation, supervision)	Ops Staff	Ops Staff	Ops Staff	Ops Staff	Systems and/or staff in OCC

Systems (including CBTC) assume responsibility for more functions

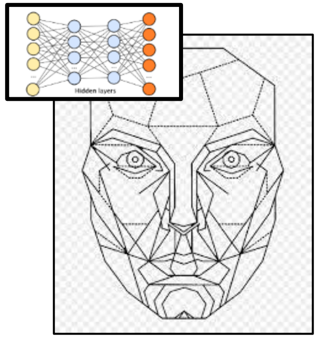
Infrastructure

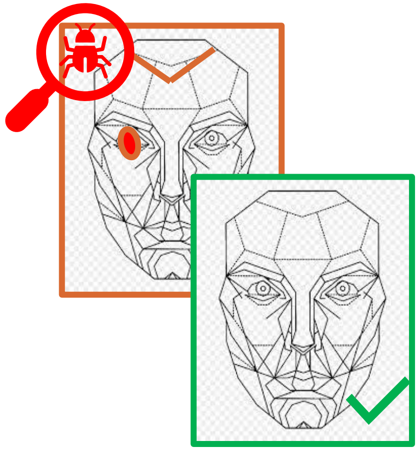
- Infrastructure monitoring
  - Broken tracks, damaged basis, …
- Positioning of train (improve location for different use cases)
  - Use less reference points like (balises - milestones for train reference points)
  - Initial position detection after “wakeup”
  - General positioning of train

Potential problems in the application of NN

potential problems in the application of NN

Data	„Ideal“ model	Model implementation (fault-free)	Model implementation (with faults)





	Data	„Ideal“ model	Model implementation (fault-free)	Model implementation (with faults)
Example problem:	**Inappropriate data, e.g. relevant samples missing**	**Incoming data noisy; Model biased, poor architecture**	**Poor SW/HW implementation**	**Platform fault during runtime**
Possible methods:	Dataset selection or augmentation, optimized training	**Uncertainty estimation (aleatoric, epistemic)** **Augmentation and dropout during training**	Network optimization (e.g. quantization)	**Platform fault-tolerance**

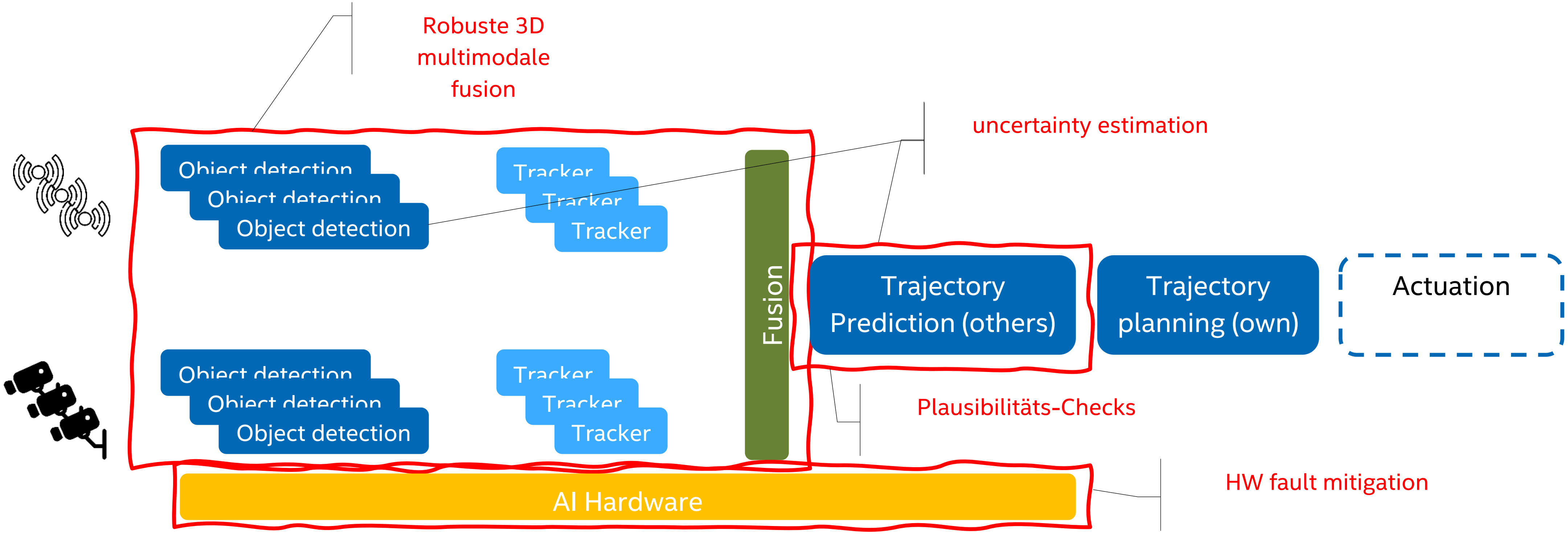
Techniques to establish resilience against platform errors, on a SW level, at inference time.
High relevance for safety-critical AI use cases, Intel use cases

options to improve the safety of NN

subtopics to cover:

- robust multimodal fusion
- HW fault mitigation
- plausibility checks
- uncertainty analysis

options to improve the safety of NN



1) Robust multi-modal fusion

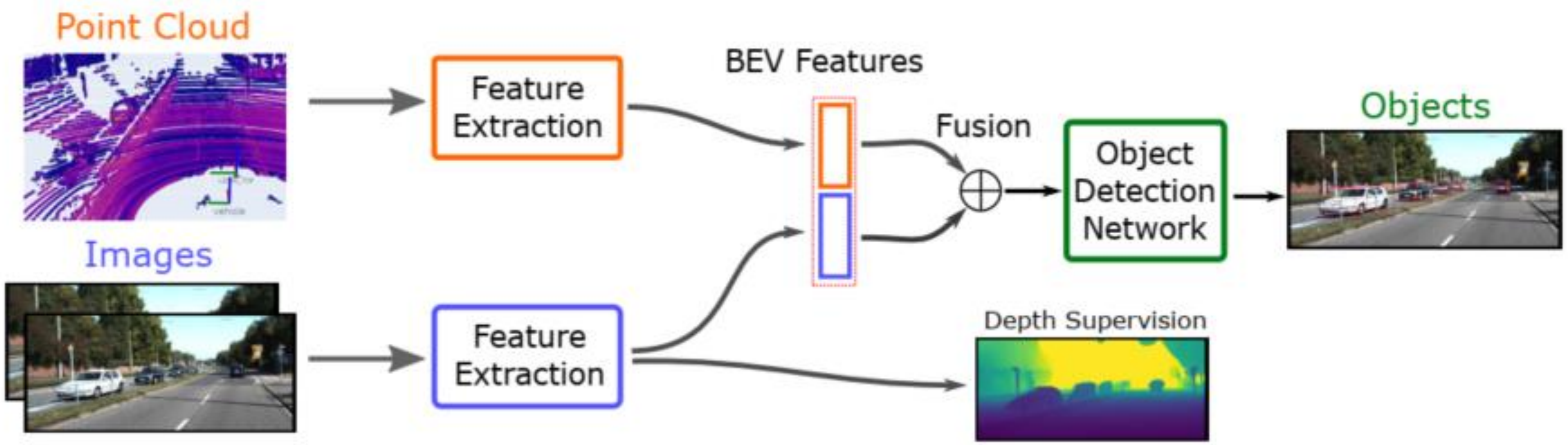
Objective: minimize the influence of faulty sensors.

Causes of faults:
- Wear
- external interferences
- Hardware or software faults

State of the Art:
- Combination of multiple sensor types improves accuracy of predictions -> methods, however, usually expect error-free sensor data
- generation of 3D object detections often only with one modality

1) Our Method

- Combination of 2 modalities camera and lidar, both of which can detect objects in 3D independently or combined. If one modality does not provide data, object detection can still be performed
- In addition, the system was made more robust by data augmentation (A) and sensor dropout (D) during training



- **J. Jarquin Arroyo et al**, "A Fault-Tolerant Multimodal 3D Object Detection Network", pending conference submission 2022.

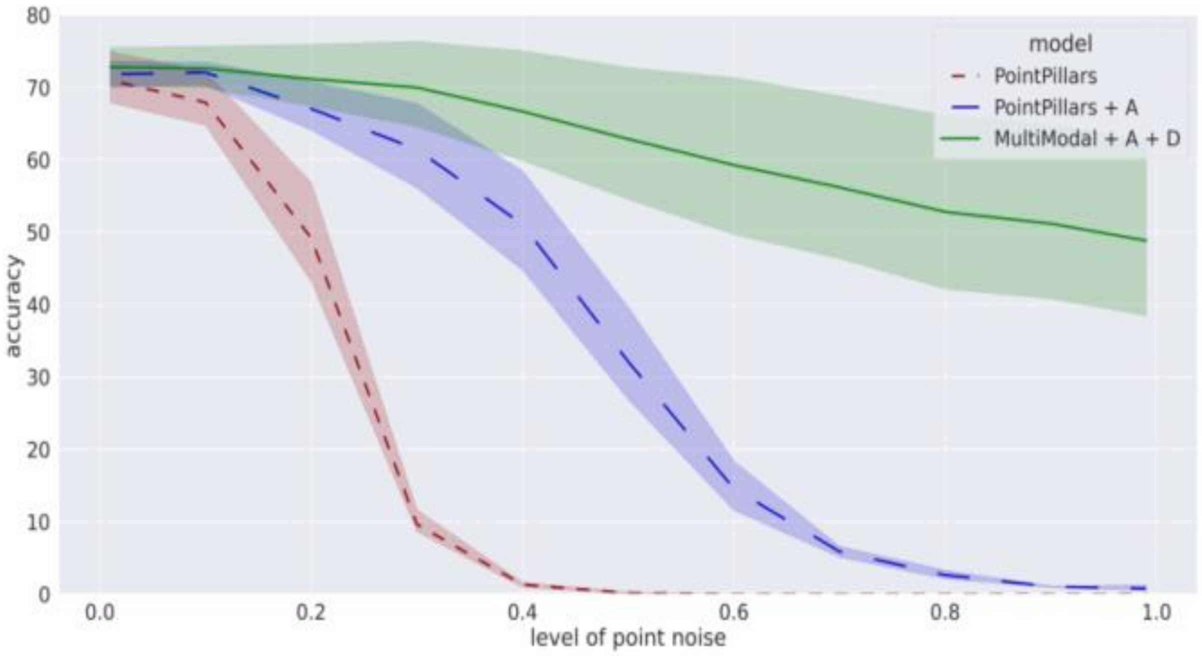
1) Train for robustness against faults

Applied Data Augmentation (A) during training

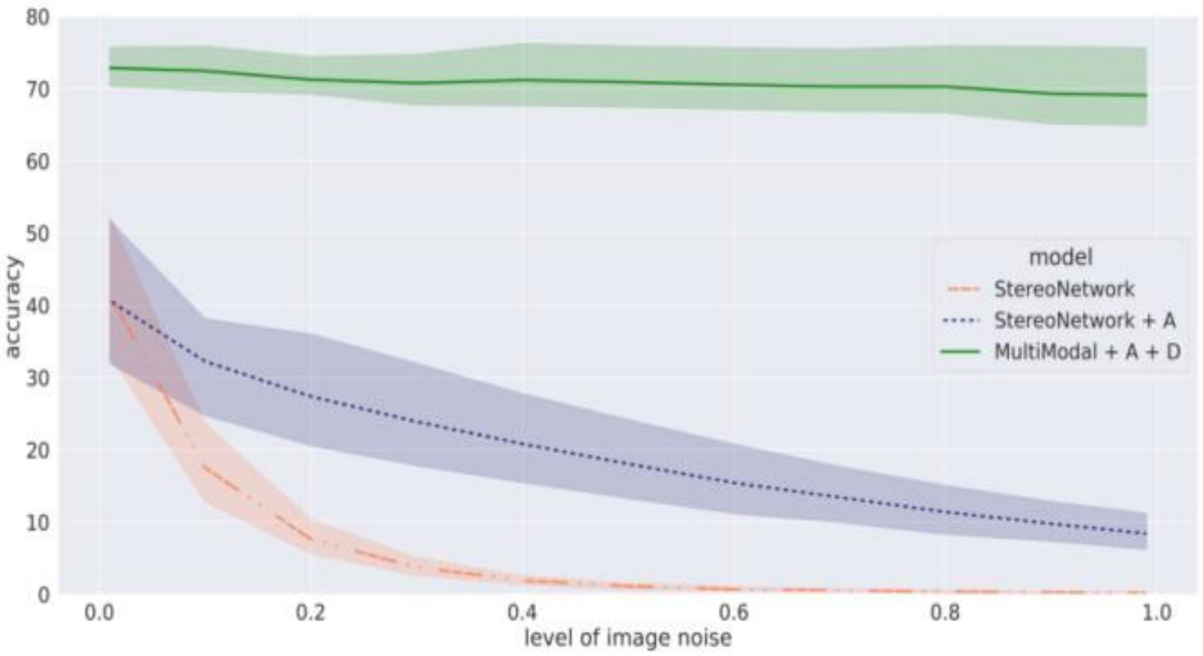
Lidar
- Point noise
- Point erasing

Camera
- Image noise
- Image erasing

1) Selected Results



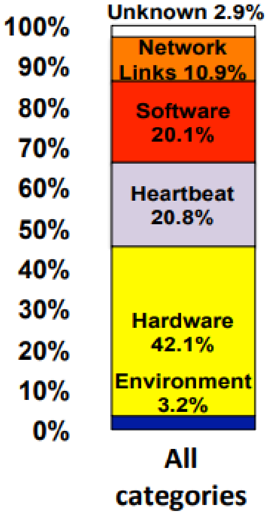
Lidar: Improvement over PointPillars and PointPillars + Augmentation



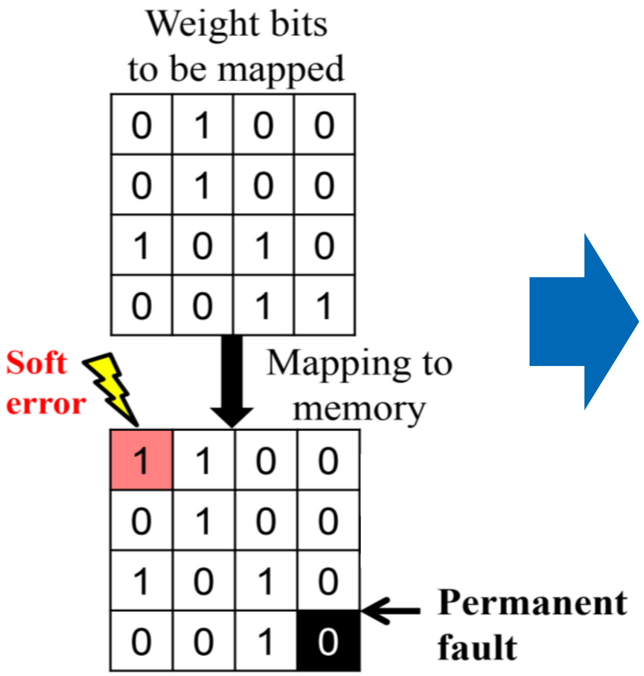
Camera: improvement over stereo network and stereo network + augmentation

2) Hardware fault mitigation

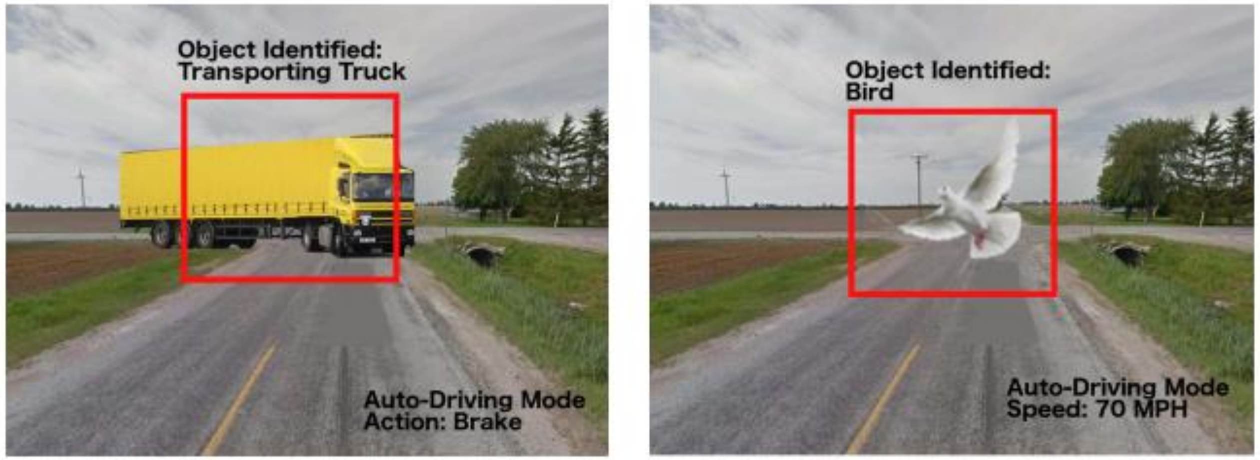
Blue Waters supercomputer
Failure root causes



Di Martino et al, 2014



Hoang et al, 2019



Li et al, 2017

Silent data corruption (SDC)

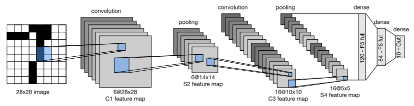
Fault model:

- Bit flip or stuck-at-0/1 faults
- Permanent/transient errors
- Network neurons (activations) or weights

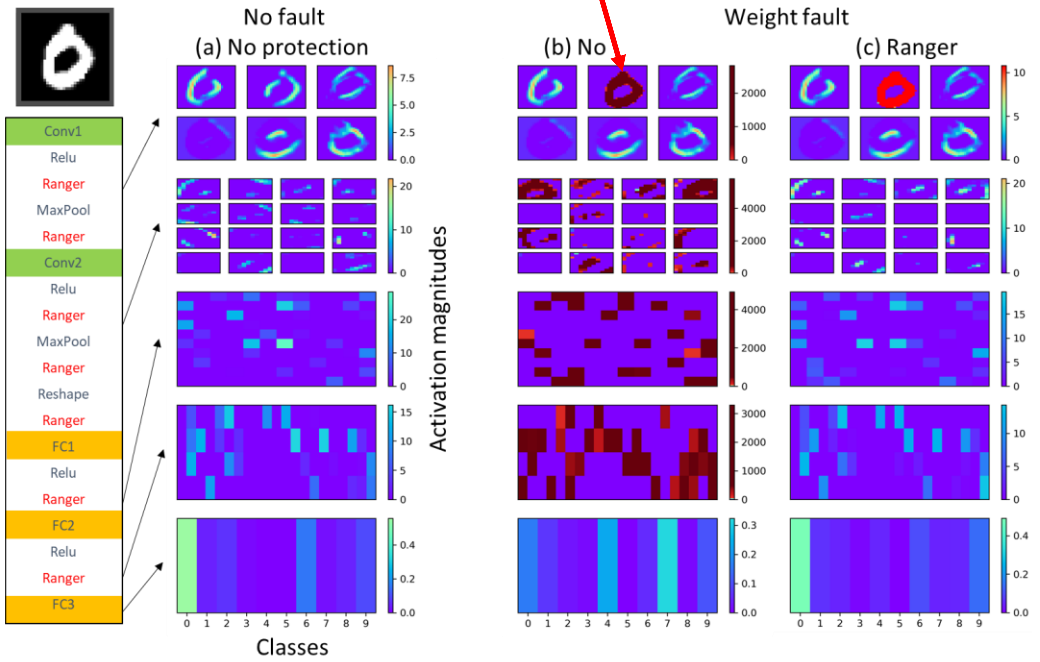
F. Geissler, Q. Syed, S. Roychowdhury, A. Asgari, Y. Peng, A. Dhamasia, R. Graefe, K. Pattabiraman, and M. Paulitsch, „Towards a Safety Case for Hardware Fault Tolerance in Convolutional Neural Networks Using Activation Range Supervision", AISafety workshop 2021

2) Error Detection & Mitigation

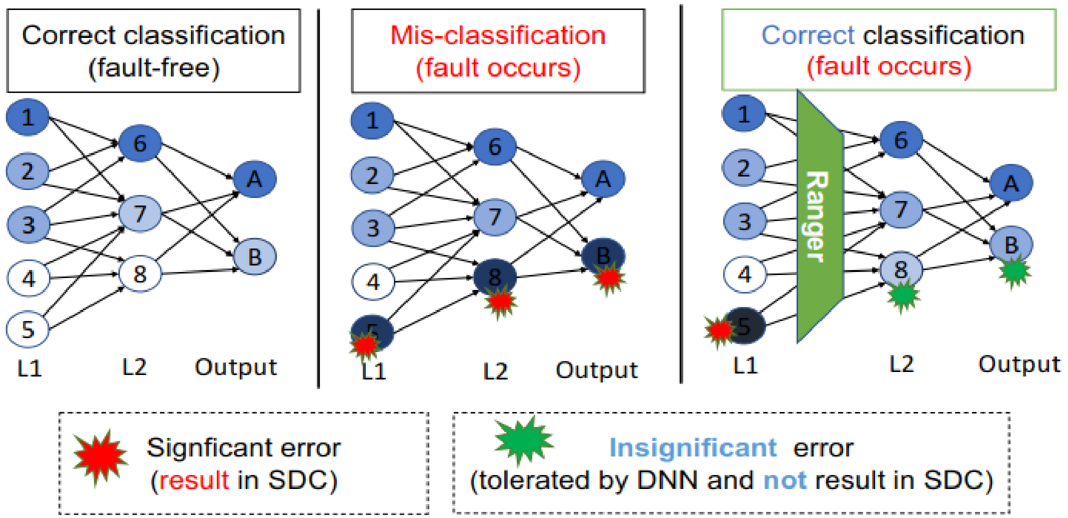
Chen et al 2020 ("Ranger")
Li et al, 2017
Geissler et al., 2021



Corrupted feature map



Chen et al, 2020



Ranger: Detect and contain faulty values by ranger monitoring and truncation

Novelties:

- Extended methods of range supervision beyond value truncation
- Fine-granular fault injection (neuron/weight, transient/permanent) using our own tool (Pytorch-ALFI)

Example: Neuron fault

<u>Setup</u>
Model: Yolov3 (pretrained on Coco)
Dataset: Custom P++ (50 images)
Injections: 1000
Fault model: Weights/Neurons, permanent stuck-at-1
Metric: SDC = change in any of TP, FP, FN
Result: SDC (No ranger → ranger):

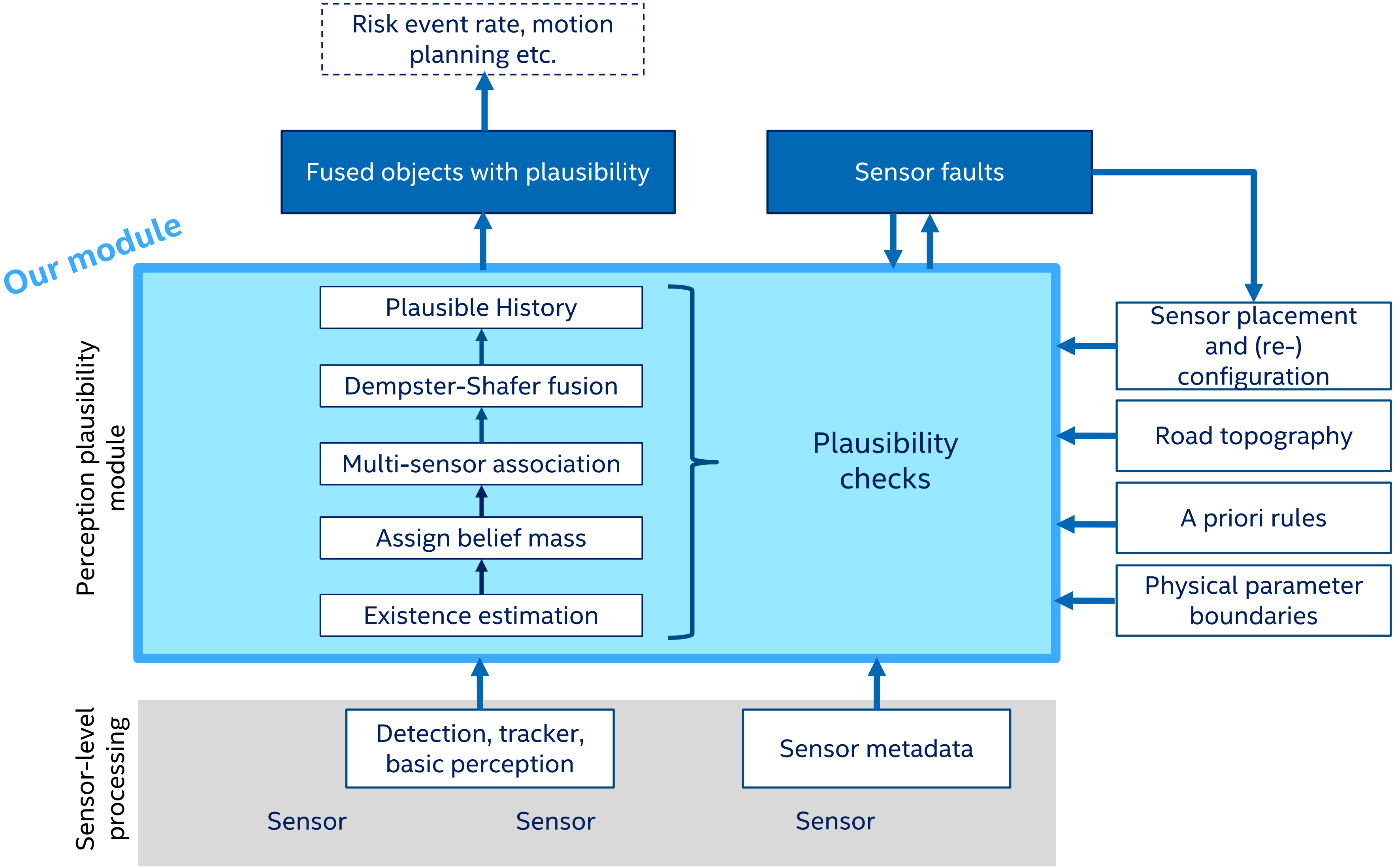
- **1.4% → 0.0%** (neurons);
- **2.7% → 0.9%** (weights)

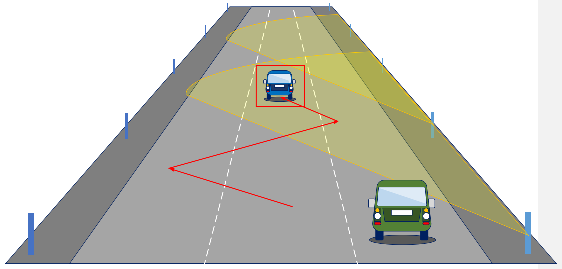
Bounding box color code

- True positives (TP)
- False positives (FP)
- False negatives (FN)

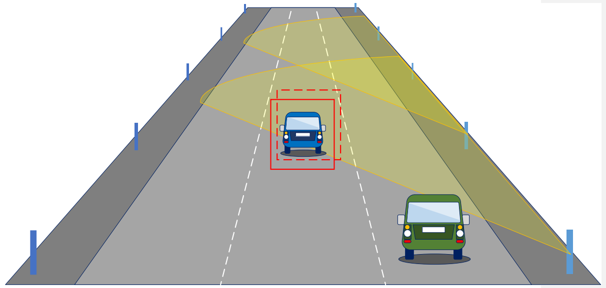
	No Ranger	With Ranger
No fault		
Fault		

3) Plausibility checking

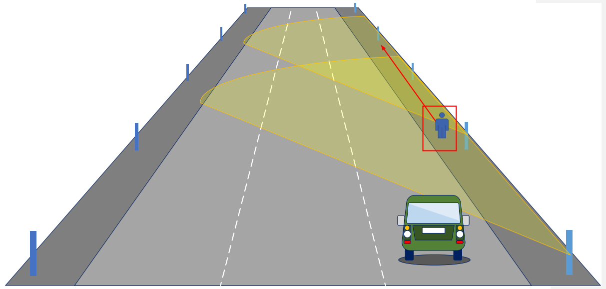




Implausible: **Object history/trajectory**



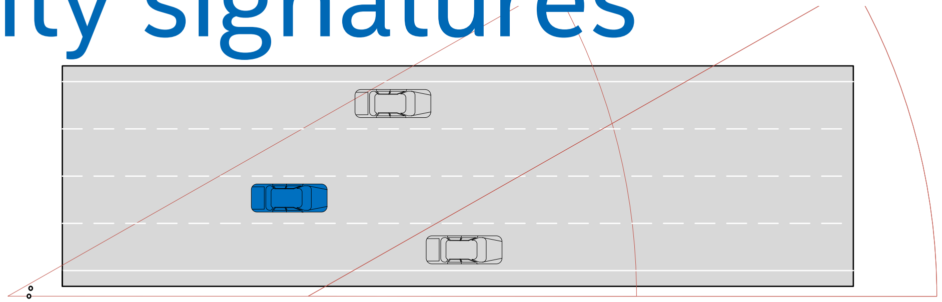
Implausible: **Inconsistent detections**



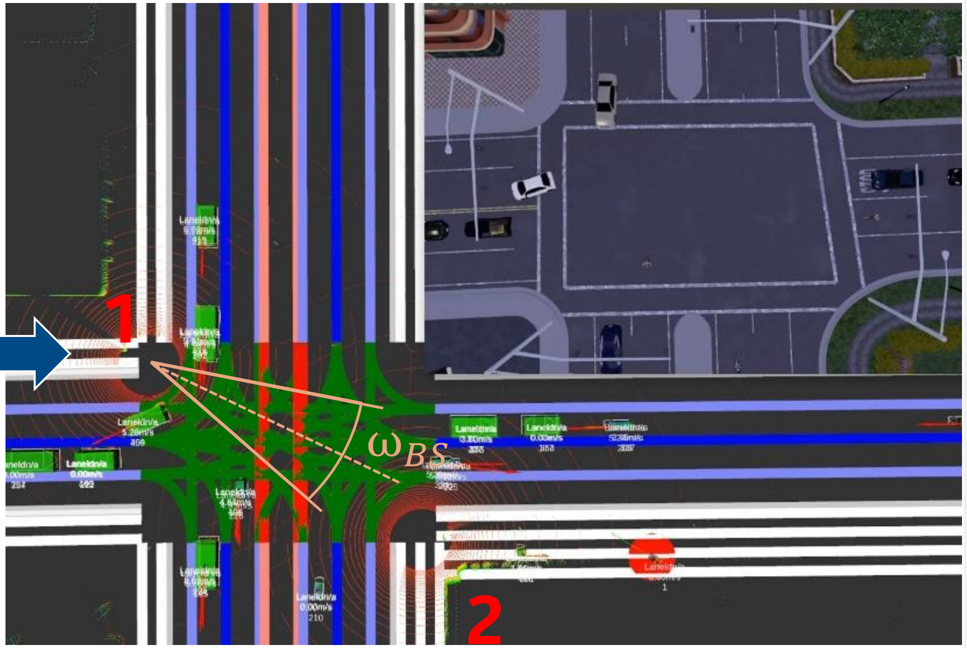
Implausible: **Object velocity w.r.t. class**

F. Geissler, A. Unnervik, and M. Paulitsch, „A Plausibility-based Fault Detection Method for High-level Fusion Perception Systems", Open Journal of Intelligent transportation systems, 2020

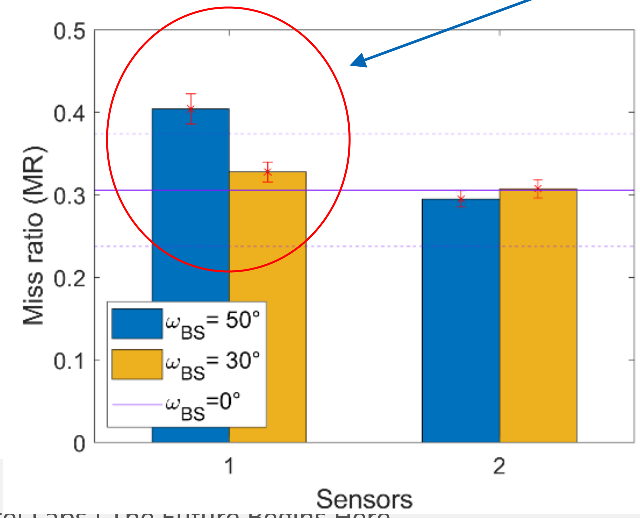
3) Example: Detecting sensor faults from plausibility signatures

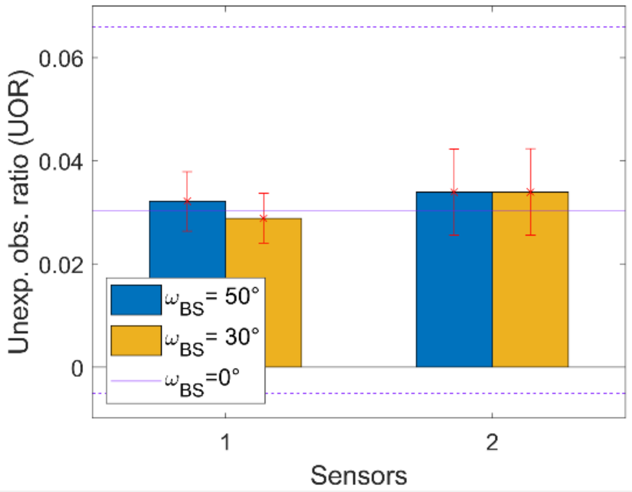


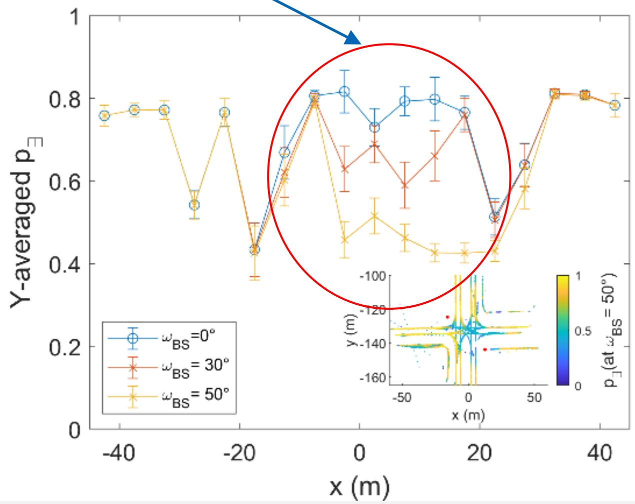
Fault: Sensor blind spot (corrupted due to dirt/dust/water etc.)



Signature: Increased miss ratio of sensor 1 (compared to 2) and reduced probability existence estimate (=plausibility score)







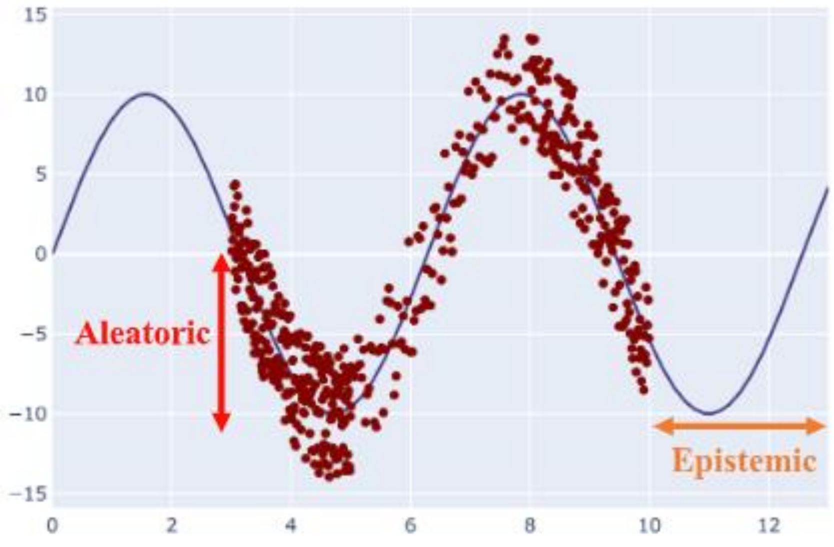
4) Uncertainty analysis – Epistemic and Aleatoric uncertainty

Aleatoric uncertainty

The irreducible uncertainty in data that gives rise to uncertainty in predictions is aleatoric uncertainty (also known as data uncertainty).This type of uncertainty is not a property of the model, but rather is an inherent property of the data distribution, and hence, it is irreducible.

Epistemic uncertainty

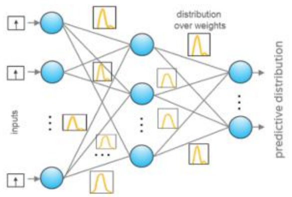
In contrast, epistemic uncertainty (also known as knowledge uncertainty) occurs due to inadequate knowledge. One can define models to answer different questions in model-based prediction.



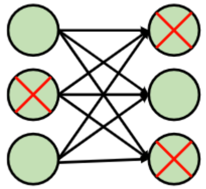
M. Abdar *et al.*, "A review of uncertainty quantification in deep learning: Techniques, applications and challenges," *Inf. Fusion*, vol. 76, pp. 243–297, Dec. 2021, doi: 10.1016/j.inffus.2021.05.008.

4) sampling based methods

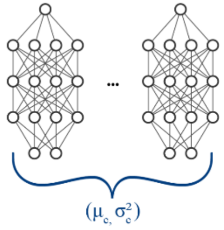
- Sampling based refers to methods where several outputs are produced for the same input using these to create a likelihood distribution e.g. a gaussian with mean and standard deviation
- The main classes of methods are:
  - Bayesian methods where static weights are replaced with distributions from which values are drawn for each pass producing slightly different results each time for the same input.[1]



  - Monte Carlo Dropout where connections or neurons are randomly dropped to create slightly different results each pass. This method is also used in training to reduce overfitting of models[2]



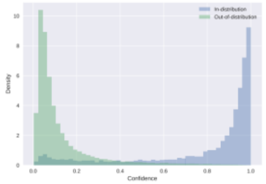
  - Deep Ensembles where slightly differently trained networks are run in parallel and produce different results in that way. Monte Carlo Dropout and Deep Ensembles can also be combined in that the different ensemble members are created by dropout.[3]

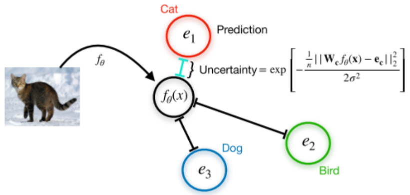


[1] C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra, "Weight Uncertainty in Neural Networks," 32nd Int. Conf. Mach. Learn. ICML 2015, vol. 2, pp. 1613–1622, May 2015, doi: 10.48550/arxiv.1505.05424.
[2] Abdar, M., Pourpanah, F., Hussain, S., Rezazadegan, D., Liu, L., Ghavamzadeh, M., Fieguth, P., Cao, X., Khosravi, A., Acharya, U. R., Makarenkov, V., & Nahavandi, S. (2021). A review of uncertainty quantification in deep learning: Techniques, applications and challenges. Information Fusion, 76, 243–297. https://doi.org/10.1016/j.inffus.2021.05.008
[3] Lakshminarayanan, B., Pritzel, A., & Blundell, C. (2016). Simple and Scalable Predictive Uncertainty Estimation using Deep Ensembles. Advances in Neural Information Processing Systems, 2017-December, 6403–6414. https://doi.org/10.48550/arxiv.1612.01474

4) sampling free methods

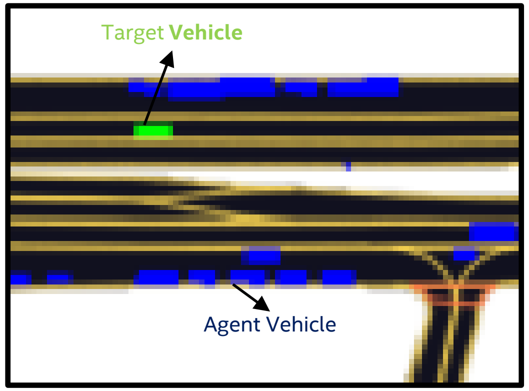
- Sampling free methods allow to create a confidence score in a single pass during inference
- The main types are:
  - **Aleatoric Uncertainty Estimation via Redundancy** uses the fact that current state-of-the-art object detectors already produce a set of object observations for the classification and their bounding boxes. In this way probability distributions for position and size of bounding boxes can be calculated in a single pass.[1]
  - **Learned Confidence Estimates** learns the confidence values by incentivising the neural network to produce confidence estimates which correctly reflect the ability of the model to produce correct predictions for given inputs in exchange for a reduction in loss.
    A variant of this is called **Loss Attenuation** where additional output vectors are appended to each anchor in object detection.[2]
  - **Deterministic Uncertainty Quantification** builds upon ideas of Radial Basis Function (RBF) networks. By enforcing detectability of changes in the input using a gradient penalty, it is able to reliably detect out of distribution data. It trains centroids per class and measures confidence scores as distance to these centroids.



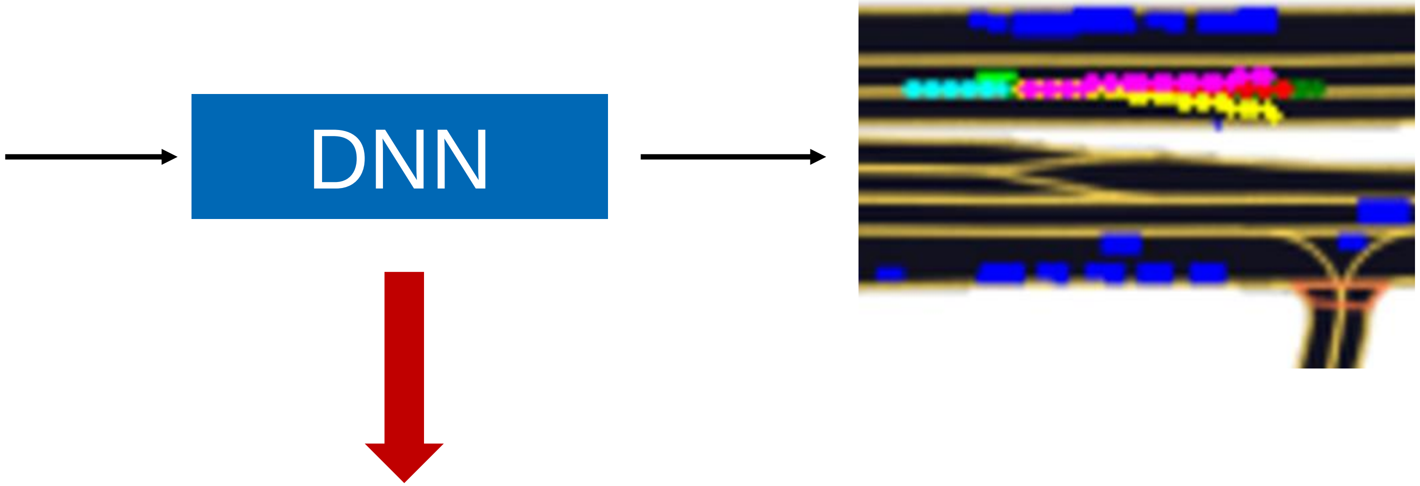


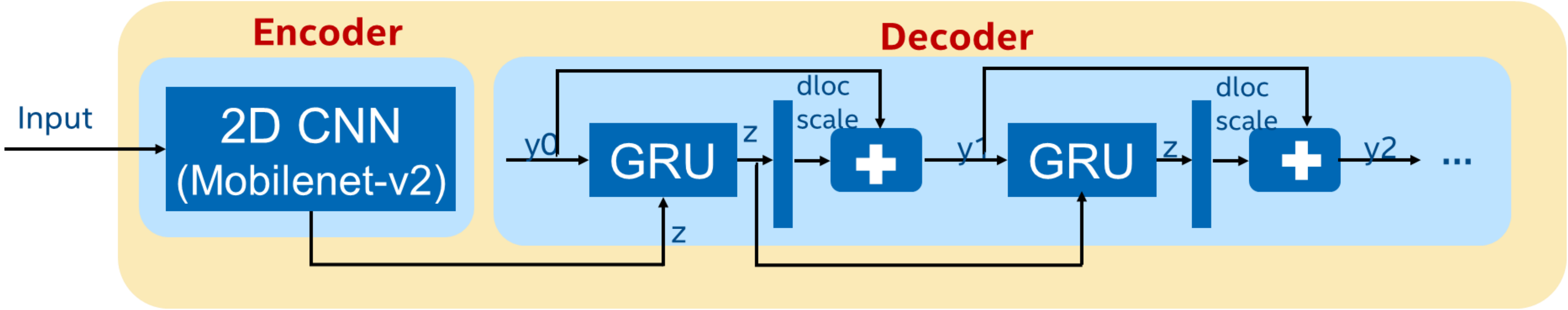
[1] Le, M. T., Diehl, F., Brunner, T., & Knoll, A. (2018). Uncertainty Estimation for Deep Neural Object Detectors in Safety-Critical Applications. IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC. https://doi.org/10.1109/ITSC.2018.8569637
[2] T. DeVries and G. W. Taylor, “Learning Confidence for Out-of-Distribution Detection in Neural Networks,” Feb. 2018, doi: 10.48550/arxiv.1802.04865.
[3] T. DeVries and G. W. Taylor, “Learning Confidence for Out-of-Distribution Detection in Neural Networks,” Feb. 2018, doi: 10.48550/arxiv.1802.04865.

4) Uncertainty-aware trajectory prediction



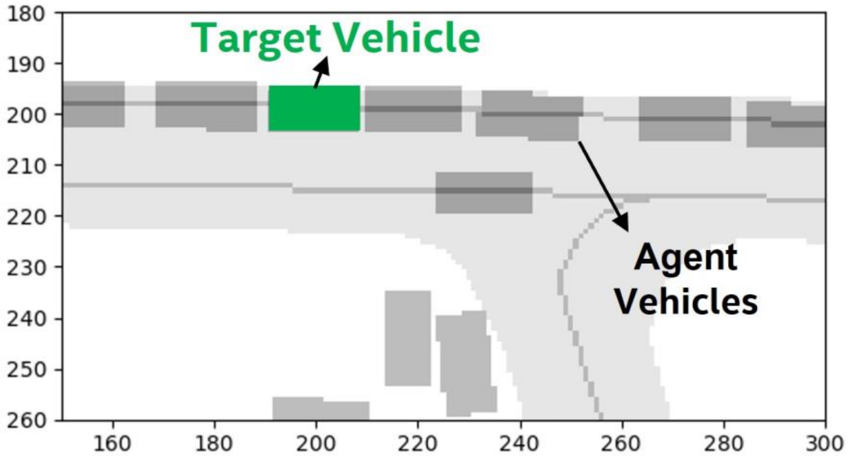
Input: Static Features with Time-dependent Features



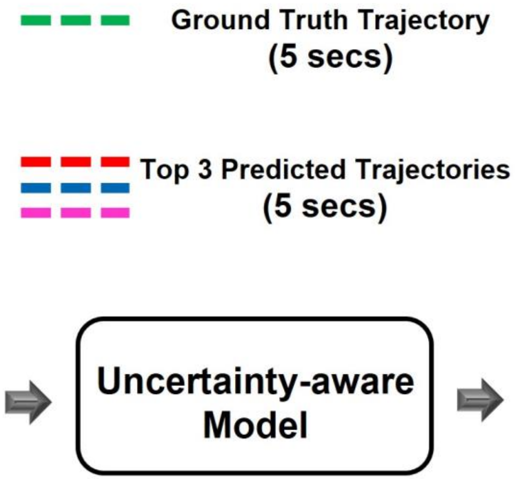


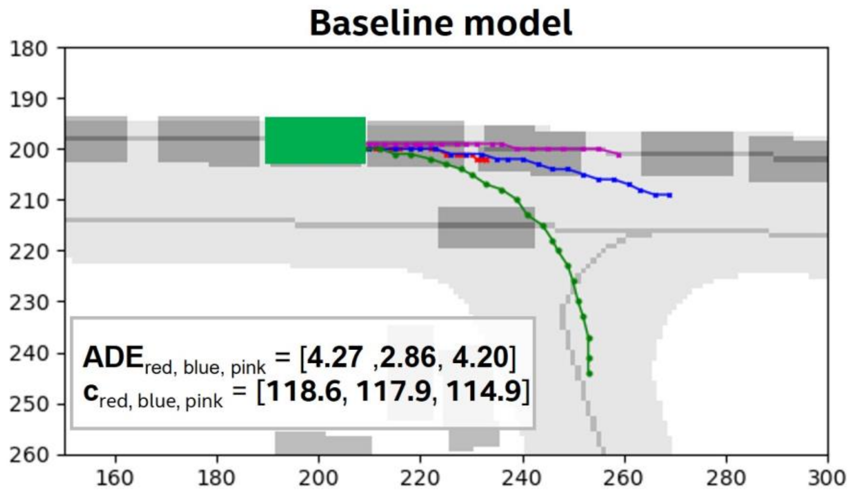
Density Estimator (Likelihood Model) **[Codevilla et al., ICRA 2018]*

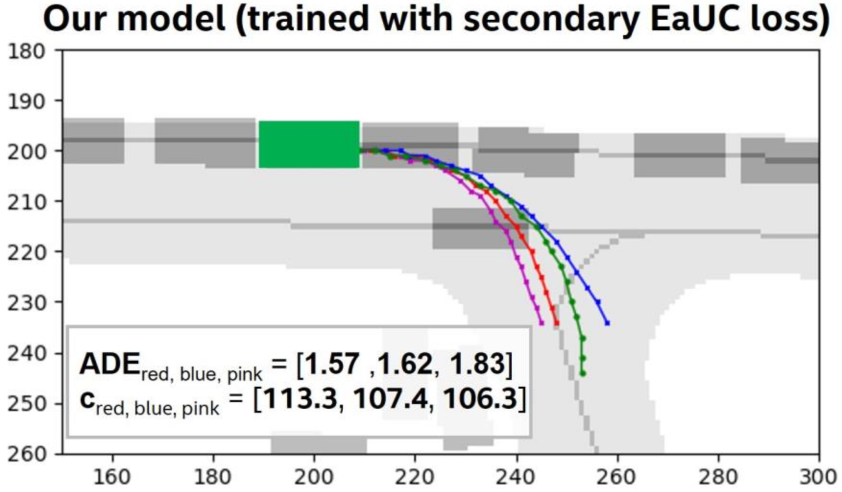
4) Error Aligned Uncertainty Optimization



Input: Static and Time-dependent Features







- **N. Kose, R. Krishnan, A. Dhamasia, O. Tickoo, M. Paulitsch**, "Error aligned uncertainty optimization to improve model calibration", will be submitted to either British Machine Vision Conference (BMVC) or European Conference on Computer Vision (ECCVW), 2022.

Thanks for you attention!

For additional question please contact

Intel Deutschland GmbH

ralf.graefe@intel.com

michael.paulitsch@intel.com

heiner.genzken@intel.com

intel®