SAFEXPLAIN

Safe and Explainable
Critical Embedded Systems based on AI

Efficient Diverse Redundant DNNs for Autonomous Driving

Martí Caro[1,2], Jordi Fornt[1,2], and Jaume Abella[1]

[1] Barcelona Supercomputing Center (BSC), Spain

[2] Universitat Politècnica de Catalunya (UPC), Spain

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH

Funded by the European Union

This project has received funding from the European Union's Horizon Europe programme under grant agreement number 101069595.

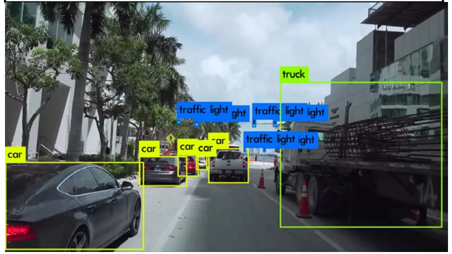
Critical Real-Time Embedded Systems (CRTES)

- CRTES are systems with **critical functionalities** that must guarantee the completion within a **deadline** and deliver **correct results**
  - Timely execution is as important as functional correctness
  - Producing a correct output after the specified deadline could lead to a **potentially fatal accident** (e.g., the ABS of a car)
- CRTES must undergo a rigorous process before being deployed to ensure meeting **safety standards**

Autonomous Driving

- Vehicles capable of making **all driving decisions**
- Mostly based on **Deep Learning algorithms**
- Stochastic process involving some **randomness and uncertainty**
- Provide **fault tolerance**
- **Perception Module:** detect the objects surrounding the vehicle
  - You Only Look Once (**YOLO**): **efficient real-time** Camera-Based Object Detector (CBOD)
- YOLOv4 is a CNN made with **162 layers** and can detect **80 classes** of objects
- We build on top of the **Darknet framework** (open source) which implements **image** and **video processing**

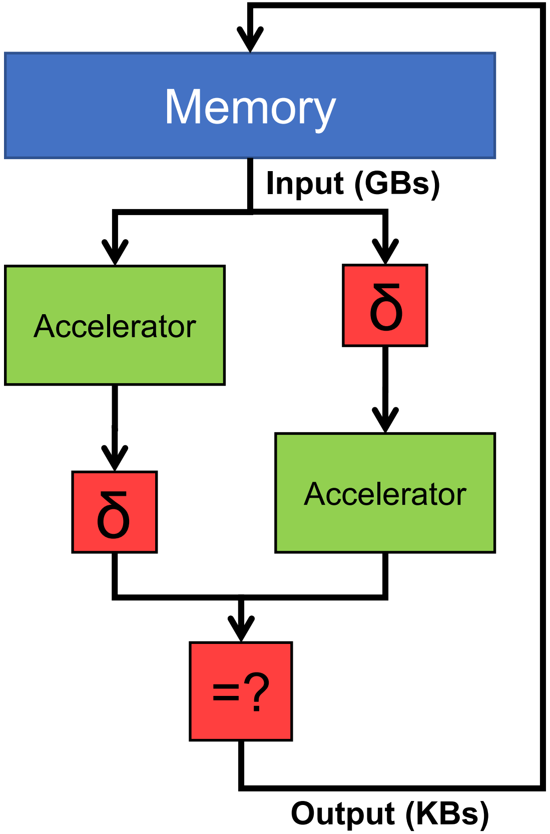
Camera-based object detection (YOLO)



SAFEXPLAIN

Motivation for Energy-Efficient Diverse Redundancy

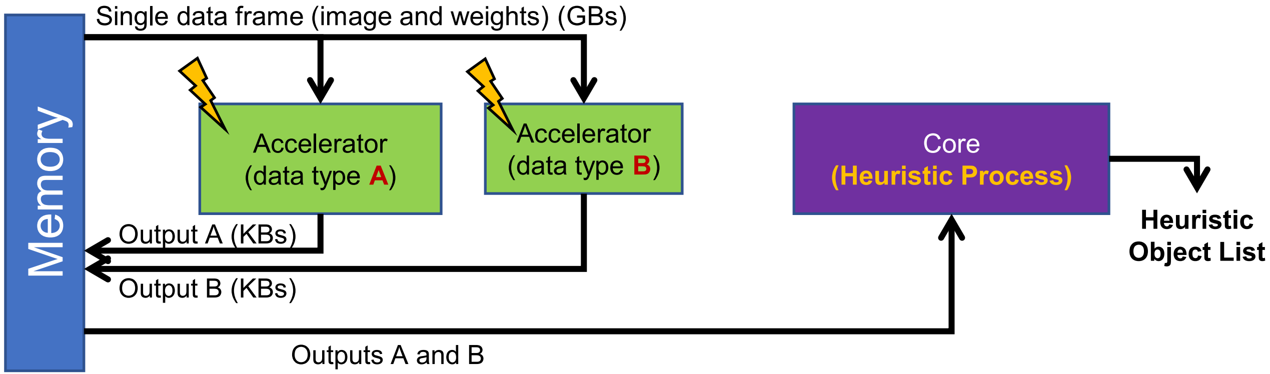
- Redundant execution often used for fault detection
- Fault tolerance for Common Cause Failures (CCFs)
  - CCFs → a single fault affects redundant copies analogously
  - **Diverse redundancy** needed
    - Different outcomes, potentially erroneous, under the same common fault
- CBOD needs to generate a **new object list every 40ms** at a rate of 25 FPS
- Such intensive computations entail **large energy and bandwidth costs** which are a **key factor** in **resource-constrained environments**
- Classic fault tolerance models (e.g., lockstep redundancy) are very **power-hungry**
- We propose an approach to provide **energy-efficient diverse redundancy** in the context of autonomous driving



SAFEXPLAIN

Diverse and Redundant Accelerators

- Our proposal:

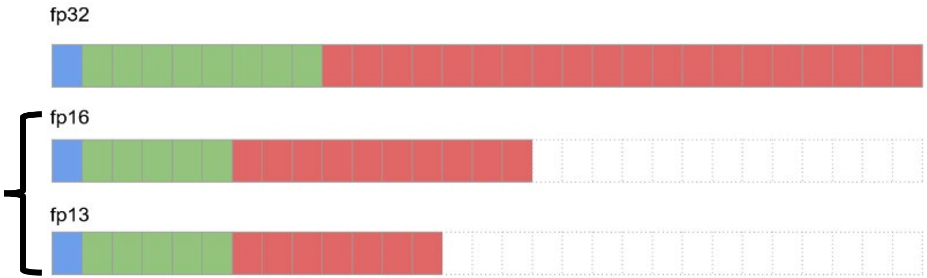


- Accelerator with data type B
  - **Lower power**
  - **Less accurate**
- Outputs A and B are **not equivalent at bit-level**
- The Heuristic compares the outputs in terms of **semantic differences** (objects detected)

SAFEXPLAIN

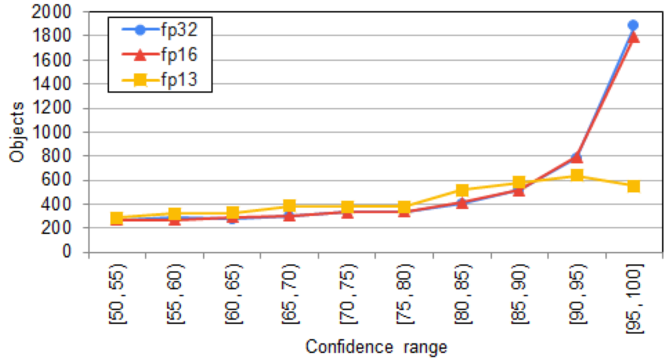
Data Types

- We have considered **FP16** as the baseline **data type A**
  - FP16 produces almost identical semantic results than FP32
  - FP8 produces unacceptable results in this case
- We have considered **FP13** as the **data type B**
  - FP13 is implemented by **dropping the 3 lowermost mantissa bits of FP16**
  - The **mantissa is the critical path** of the floating-point operations
  - Using smaller mantissas brings several benefits
    - **Shorter critical path:** Fewer bits are operated
    - **Lower energy consumption:** Lower power gates can be used to fit the shorter critical path
    - **Lower area requirements**



Confidence Values of FP13

- We show how many **objects** are **detected within each confidence range** for the COCO dataset
- Differences between fp32 and fp16 are tiny. However, this is not the case for fp13
- Objects identified with high confidence have values close to 1, but **strictly below 1**
- The highest value strictly below 1 (**HVSB1**) is farther away from 1 for lower precision arithmetic
  - HVSB1_fp16 = 0.9995, HVSB1_fp13 = 0.996
  - $HVSB1\_fp16^{20}$ = 0.990, $HVSB1\_fp13^{20}$ = 0.926
- The usual case is that $\boldsymbol{Conf^{fp16} > Conf^{fp13}}$



Error Injection Emulation

- **SoftFloat library** to emulate FP16 and FP13 at the software level
- Errors injected in the **sign or exponent** of the result of multiplication and addition operations based on a given probability (set to 10^{-10})
- The impact of faults in the **mantissa** is often completely negligible, making the fault injection campaign **highly ineffective**
- We have analysed two cases
  - **Same frame:** Errors in the same frame on both accelerators (in random operation and in the same operation)
  - **Independent Faults:** Errors in different frames

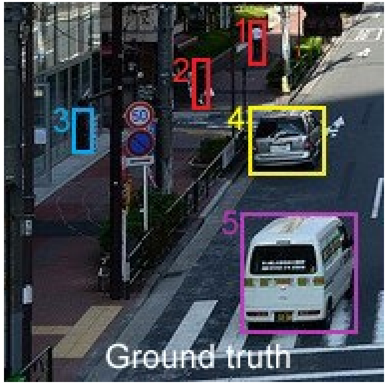
SAFEXPLAIN

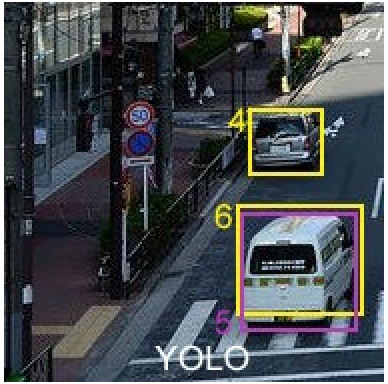
Accuracy Metric

- We have used the **Intersection over Union** (IoU) to assess the accuracy impact

$$IoU = \frac{Area_{gt} \cap Area_p}{Area_{gt} \cup Area_p}$$

- IoU ≥ t (t = 0.5)
- Predictions are classified as
  - **True Positives (TP)**
  - **False Negatives (FN)**
  - **False Positives (FP)**





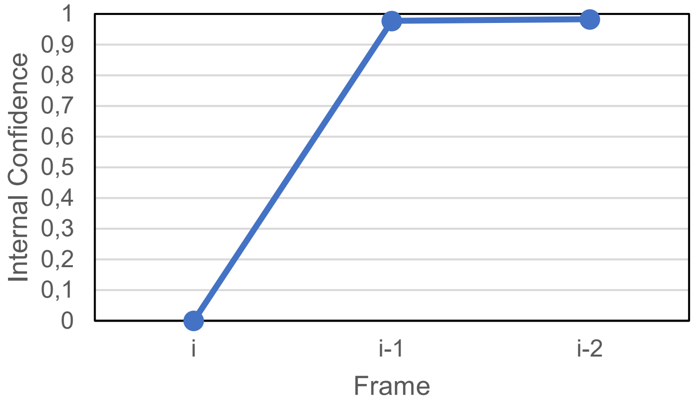
Dataset

- **Real driving videos**
  - ↑ Real data for autonomous driving
  - ↑ Average the detections of consecutive frames
  - ↓ FP16 is used as the reference model but it is not always correct
  - ↓ Need to perform visual inspection to check for true errors
- **Three sets of videos of 6 videos each**
  - **SET1train:** to test and fine-tune our scheme
  - **SET1eval:** to evaluate our scheme using the same videos but forwarded enough to grant independence
  - **SET2eval:** to evaluate our scheme with different videos that grant further independence

SAFEXPLAIN

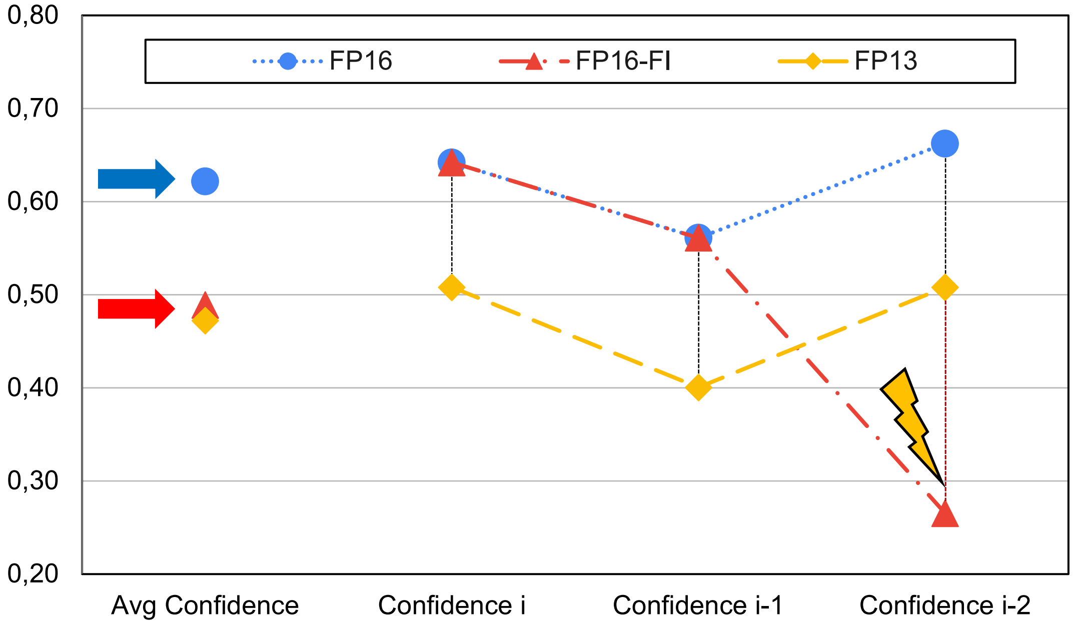
Frame Average Calculation

- **Averaging** the confidence **across 3 frames**
- The average confidence does not correspond to the direct average of the three confidences
- Example:
  - Expected Average = (0 + 0,97 + 0,98)/3 = 0,65
  - Actual Average = 0,44
- The confidence is calculated as
  - **Confidence = ObjProb × ClassProb**
- The ClassProb and ObjProb are the ones being averaged rather than the resulting confidence
  - $AvgConf = \frac{OP_{i\_2}, OP_{i\_1}, OP_i}{3} \cdot \frac{CP_{i\_2}, CP_{i\_1}, CP_i}{3}$
- The **impact** of a fault is **up to quadratic** on the average confidence



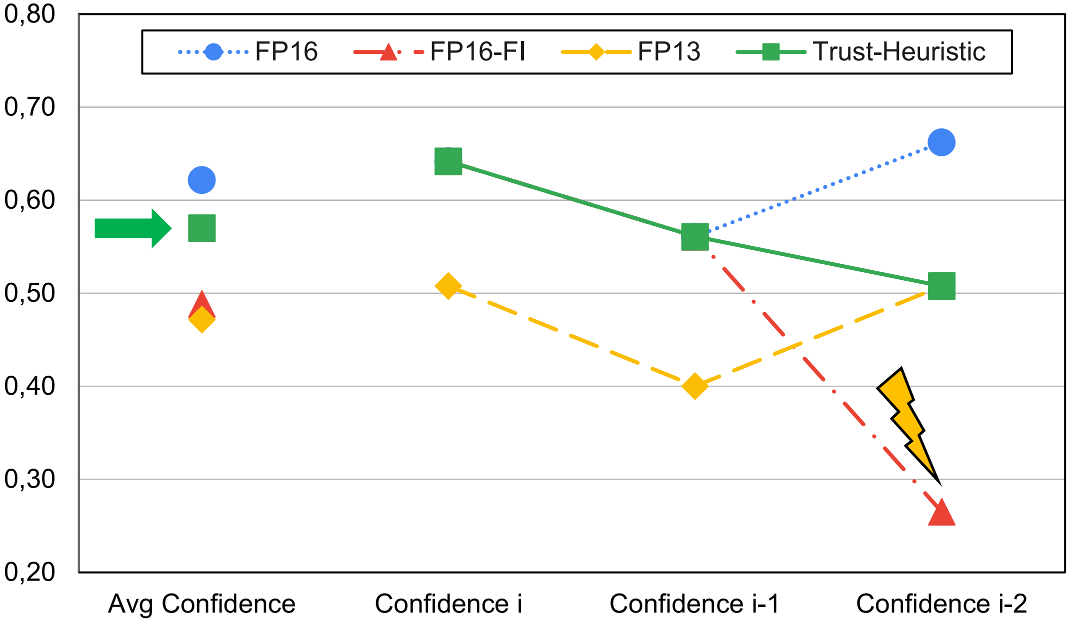
SAFEXPLAIN

Example of a Faulty Detection



- We note that the **confidence difference** between fp16 and fp13 **remain mostly stable across frames** in fault-free cases
- We **compare the confidence difference** between both accelerators for the three frames
- We regard a detection as **faulty** if one of the differences is **significantly larger** than the other two

Heuristics to Correct Faults: TRUST



- **Trust the maximum confidence** for the faulty frame

TRUST Heuristic

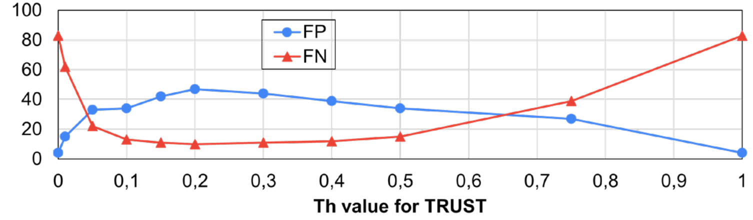
$DIF_i = |Conf_i^{fp16} - Conf_i^{fp13}|$
$DIF_{i-1} = |Conf_{i-1}^{fp16} - Conf_{i-1}^{fp13}|$
$DIF_{i-2} = |Conf_{i-2}^{fp16} - Conf_{i-2}^{fp13}|$

$COND_i = (|DIF_i - DIF_{i-1}| > Th \ \&\& \ |DIF_i - DIF_{i-2}| > Th)$
$COND_{i-1} = (|DIF_{i-1} - DIF_i| > Th \ \&\& \ |DIF_{i-1} - DIF_{i-2}| > Th)$
$COND_{i-2} = (|DIF_{i-2} - DIF_i| > Th \ \&\& \ |DIF_{i-2} - DIF_{i-1}| > Th)$

Keep OP_x^{fp16} and CP_x^{fp16} For the three frames

if $(COND_i \ \&\& \ \overline{COND_{i-1}} \ \&\& \ \overline{COND_{i-2}})$ {
  if $(Conf_i^{fp13} > Conf_i^{fp16})$
    Replace OP_i^{fp16} and CP_i^{fp16} by OP_i^{fp13} and CP_i^{fp13}
} else if $(\overline{COND_i} \ \&\& \ COND_{i-1} \ \&\& \ \overline{COND_{i-2}})$ {
  if $(Conf_{i-1}^{fp13} > Conf_{i-1}^{fp16})$
    Replace OP_{i-1}^{fp16} and CP_{i-1}^{fp16} by OP_{i-1}^{fp13} and CP_{i-1}^{fp13}
} else if $(\overline{COND_i} \ \&\& \ \overline{COND_{i-1}} \ \&\& \ COND_{i-2})$ {
  if $(Conf_{i-2}^{fp13} > Conf_{i-2}^{fp16})$
    Replace OP_{i-2}^{fp16} and CP_{i-2}^{fp16} by OP_{i-2}^{fp13} and CP_{i-2}^{fp13}
}

- TRUST with independent faults (SET1train)



- TRUST is **highly insensitive** to the value of *Th*
- We use ***Th = 0.1*** since it gives slightly better results
- Analogous results with the other sets of videos and configurations

SAFEXPLAIN

Effectiveness of TRUST

SETUP	SET1train			SET1eval			SET2eval		
	FP	FN	% fixed	FP	FN	% fixed	FP	FN	% fixed
Baseline	7	210	–	4	243	–	6	250	–
TRUST (indep. faults)	34	13	78.3%	19	38	76.9%	21	35	78.1%
TRUST (same frame)	55	60	47.0%	22	50	70.9%	23	57	68.8%

- **Most faults are corrected** (between 76.9% and 78.3%) when they affect a single accelerator
- It **reaches 94%** if we **focus on objects with confidence values above 60%**
- This occurs mostly **removing FNs**, and the vast majority of **FPs** are true objects whose **confidence** was **slightly under the threshold**
- Whenever faults are **injected synchronously**, results are naturally slightly worse

SAFEXPLAIN

Breakdown Across Small and Large Confidence Range Change

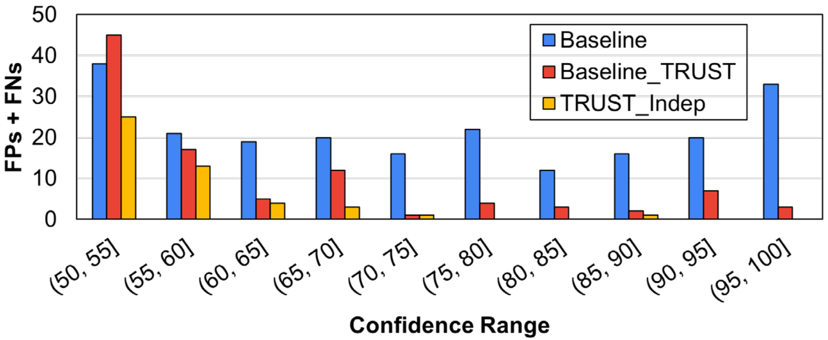
SETUP	SET1train			SET1eval			SET2eval		
	FP	FN	% fixed	FP	FN	% fixed	FP	FN	% fixed
Baseline	7	210	–	4	243	–	6	250	–
TRUST (indep. faults)	34	13	78.3%	19	38	76.9%	21	35	78.1%
TRUST (same frame)	55	60	47.0%	22	50	70.9%	23	57	68.8%

SETUP	SM	LG	% LG fixed	SM	LG	% LG fixed	SM	LG	% LG fixed
Baseline	24	193	–	16	231	–	36	278	–
TRUST (indep. faults)	15	32	83.4%	9	48	79.2%	21	35	87.4%
TRUST (same frame)	27	88	54.4%	13	59	74.5%	29	51	81.7%

- We provide results breaking down **accumulated FPs+FNs** into
  - Small changes (**SM**): Objects moving from 40%-50% confidence to 50%-60% or vice versa
  - Large changes (**LG**): Objects with larger confidence range change
- **Most LG errors are fixed** when TRUST is applied
- Whenever faults are **injected synchronously**, results are naturally slightly worse

SAFEXPLAIN

Uncorrected Errors per Confidence Range



- We analyse how many FPs and FNs **escape** in total **across different confidence ranges**
- Three configurations
  - **Baseline:** non-redundant accelerator with faults
  - **Baseline_TRUST:** redundant accelerator with all faults injected in fp16
  - **TRUST_Indep:** half of the faults injected in each of the accelerators
- TRUST corrects most of the errors with a **high confidence**
- These results are analogous on the other configurations/datasets

Accuracy Comparison Against DCLS-like Solutions

- A simple solution – yet **costly** – would be to simply **replicate the primary accelerator**
- **TRUST**, whose **cost is lower**, causes:
  - **34 FPs and 13 FNs** (SET1train), **19 FPs and 38 FNs** (SET1eval), and **21 FPs and 35 FNs** (SET2eval)
- The baseline **fp16** implementation **may also cause FPs and FNs** since we lack labelled datasets for the videos
- We have performed **visual inspection** and found out the following

DATASET	TP	FP	FN	TN
SET1train	31	3	13	0
SET1eval	17	2	38	0
SET2eval	17	4	35	0

- TRUST performs **slightly better** than DCLS with **SET1train** (16 vs 31 errors)
- TRUST performs **slightly worse** than DCLS with **SET1eval and SET2eval** (40 vs 17 errors, and 39 vs 17 errors) (**0.29-0.46%** misdetection increase)

SAFEXPLAIN

Energy Comparison Against DCLS-like Solutions

SETUP	Energy (mJ)				Δ Energy	
	OPs	DRAM	SRAM	Total	OPs	Total
FP16-only	102	788	3	893	-	-
FP13-only	72	788	3	863	-29%	-3%
DCLS-full	204	788	3	995	-	-
TRUST-full	174	788	5	967	-14%	-3%
DCLS-clus	204	263	3	469	-	-
TRUST-clus	174	263	5	442	-14%	-6%

- TRUST provides a **14% OP energy reduction**
- **3% total energy reduction**
- **Bandwidth optimizations** (e.g., weight clustering) should be applied to obtain **higher total energy savings**

SAFEXPLAIN

Conclusions and Future Work

- **Efficient diverse redundancy** becomes **mandatory for DNNs**, which are the most power-hungry computing kernel of object detection in AD
- **Full duplication** of accelerators brings **significant costs in terms of power**
- We present **TRUST**, a scheme to build diverse redundant accelerators based on the use of lower precision arithmetic to **reduce costs**, while **preserving performance** and **reusing data fetched** by the primary accelerator
- Our analysis shows that such strategy provides **effective error correction**, particularly for the most significant errors, with **3-6% energy reductions** w.r.t. DCLS-like solutions
- Part of our ongoing future work is realising such a scheme in an actual diverse and redundant accelerator exploiting the findings in this paper and evaluating TRUST with other arithetmics (e.g., integer)

SAFEXPLAIN

THANKS

Contact: Martí Caro

Email: marti.caroroca@bsc.es

SAFEXPLAIN

Safe and Explainable
Critical Embedded Systems based on AI

GOBIERNO DE ESPAÑA MINISTERIO DE CIENCIA E INNOVACIÓN

This work has been partially supported by the Spanish Ministry of Science and Innovation under grant PID2019-107255GBC21/AEI/10.13039/501100011033

Follow us on social media:

www.safexplain.eu

Funded by the European Union

This project has received funding from the European Union's Horizon Europe programme under grant agreement number 101069595.