Transfer Learning and Hyperparameter Optimization for Automatic Car Damage Detection

UNIVERSITÀ DEGLI STUDI DI MILANO BICOCCA

Data Science Master Course

Department of Informatics, System and Communication

Academic Year 2019/2020

Supervisor: Antonio Candelieri

Candidate: Raffaele Anselmo

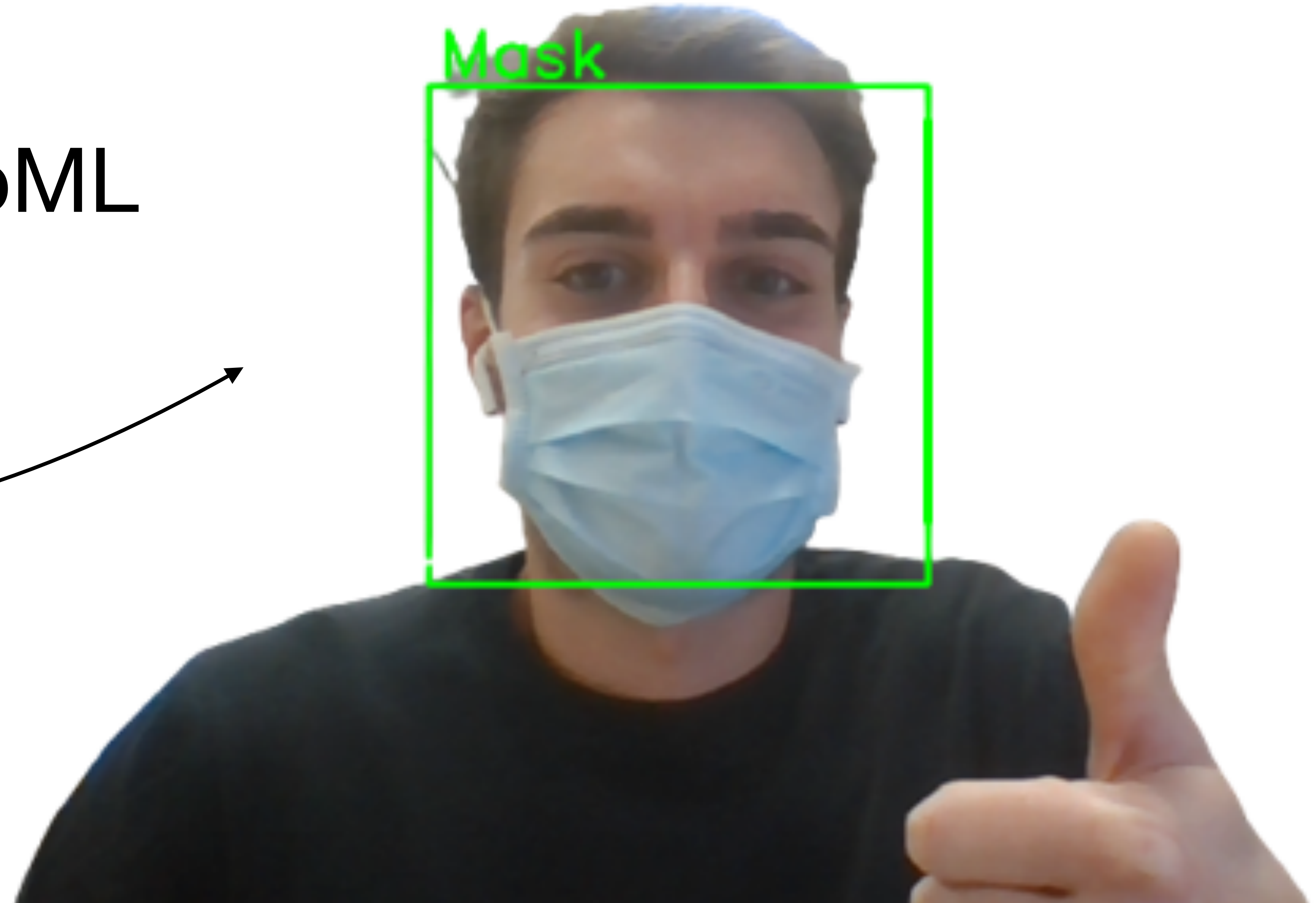
846842

Who am I?

Bachelor Degree in Statistical Sciences @Unibo

Currently Data Scientist @CRIF

DS favourite topics: CV, Fair ML and AutoML

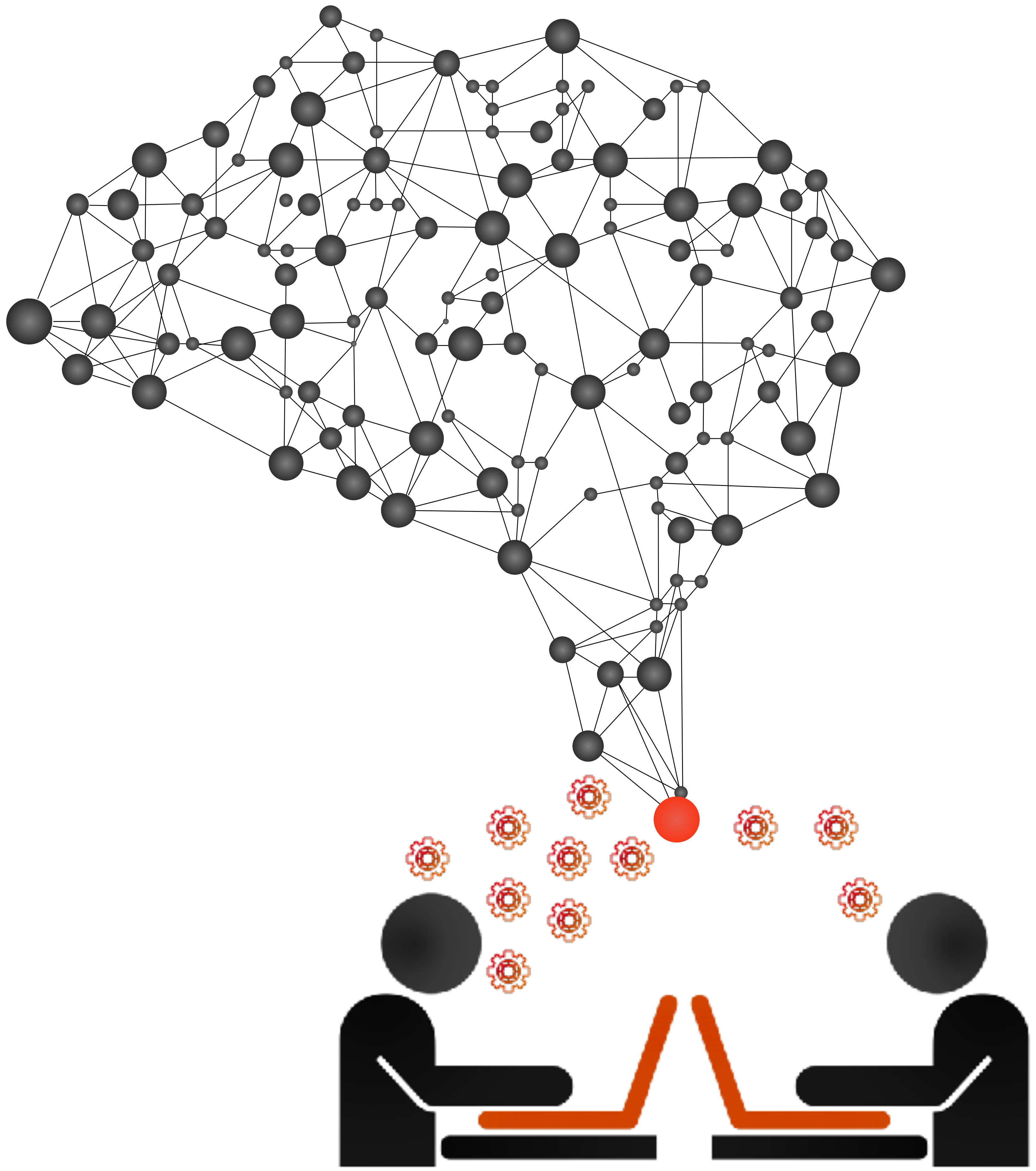


What is this Thesis about?

Hyperparameters Optimization
TASK

on Transfer Learning
MODEL PIPELINE

for automatic car Damage Detection
APPLICATION



How is it structured?

1 Image Classification

From Hubel & Wiesel experiment to Alexnet and beyond

2 Hyperparameters Optimization

AutoML, Bayesian Optimization, Hyperband, BOHB

3 Damage Detector

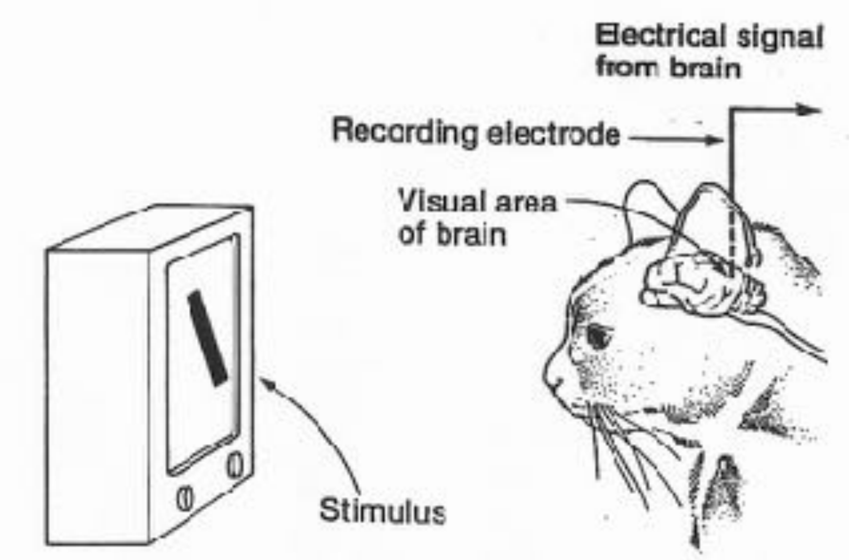
A business solution for insurance market

Image Classification

BACKGROUND

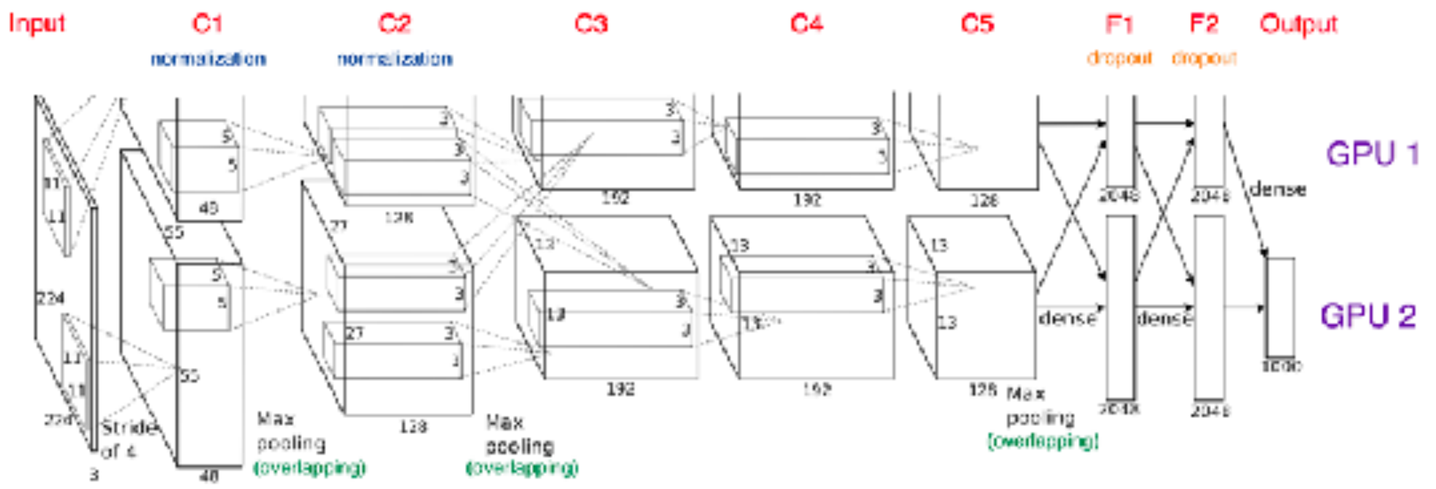
LeNet

Hubel & Wiesel Experiment



Fukushima's Necognitron

Alexnet: the breakthrough



1950 1960 1970 1980 1990 2000 2010 2020

Hyperparameters Optimization

AutoML

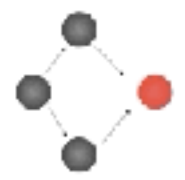
Automate all the design decisions in a data-driven, objective, and automated way

HPO

Automate the search for an optimal predictive model

NAS

Automate the architecture design of ANNs



Hyperparameters Optimization

AutoML

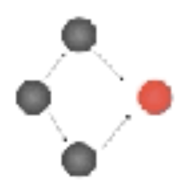
Automate all the design decisions in a data-driven, objective, and automated way

HPO

Automate the search for an optimal predictive model

NAS

Automate the architecture design of ANNs



Hyperparameters Optimization

HPO

Automate the search for an optimal predictive model

$$\lambda^* = \arg\min_{\lambda \in \Lambda_N} \mathbf{V}(\mathcal{L}, \mathcal{A}_\lambda, D_{train}, D_{val})$$

Where:

$\lambda \in \Lambda_N$ = vector of hyperparameters

$V(\cdot, \cdot, \cdot, \cdot)$ = validation protocol

A_λ = Model A with hyperparameters set λ

$\mathscr{L}$ = Loss of the model

Hyperparameters Optimization

HYPERBAND

Algorithm 2: Hyperband algorithm using Successive Halving

Input : R, η

Initialization: $s_{max} = \lfloor \log_\eta(R) \rfloor$, $B = (s_{max} + 1)R$

for $s \in \{s_{max}, s_{max-1}, \ldots, 0\}$ **do**

  $n = \lceil \frac{B}{R} \frac{\eta^s}{(s+1)} \rceil$, $r = R\eta^{-s}$

  // begin SuccessiveHalving with (n, r) *inner loop*

  $T =$ *sample* n *configurations*

  for $i \in \{0, \ldots, s\}$ **do**

    $n_i = \lfloor n\eta^{-i} \rfloor$

    $r_i = r\eta^i$

    $L =$ *validation loss* $\{(t, r_i) : t \in T\}$

    $T = top_k(T, L, \lfloor n_i/\eta \rfloor)$

Return : *Configuration with the smallest validation loss so far*

Inner Loop
or Bracket

Outer Loop

Successive Halving + Hyperband

Strong anytime performance
Computational costs

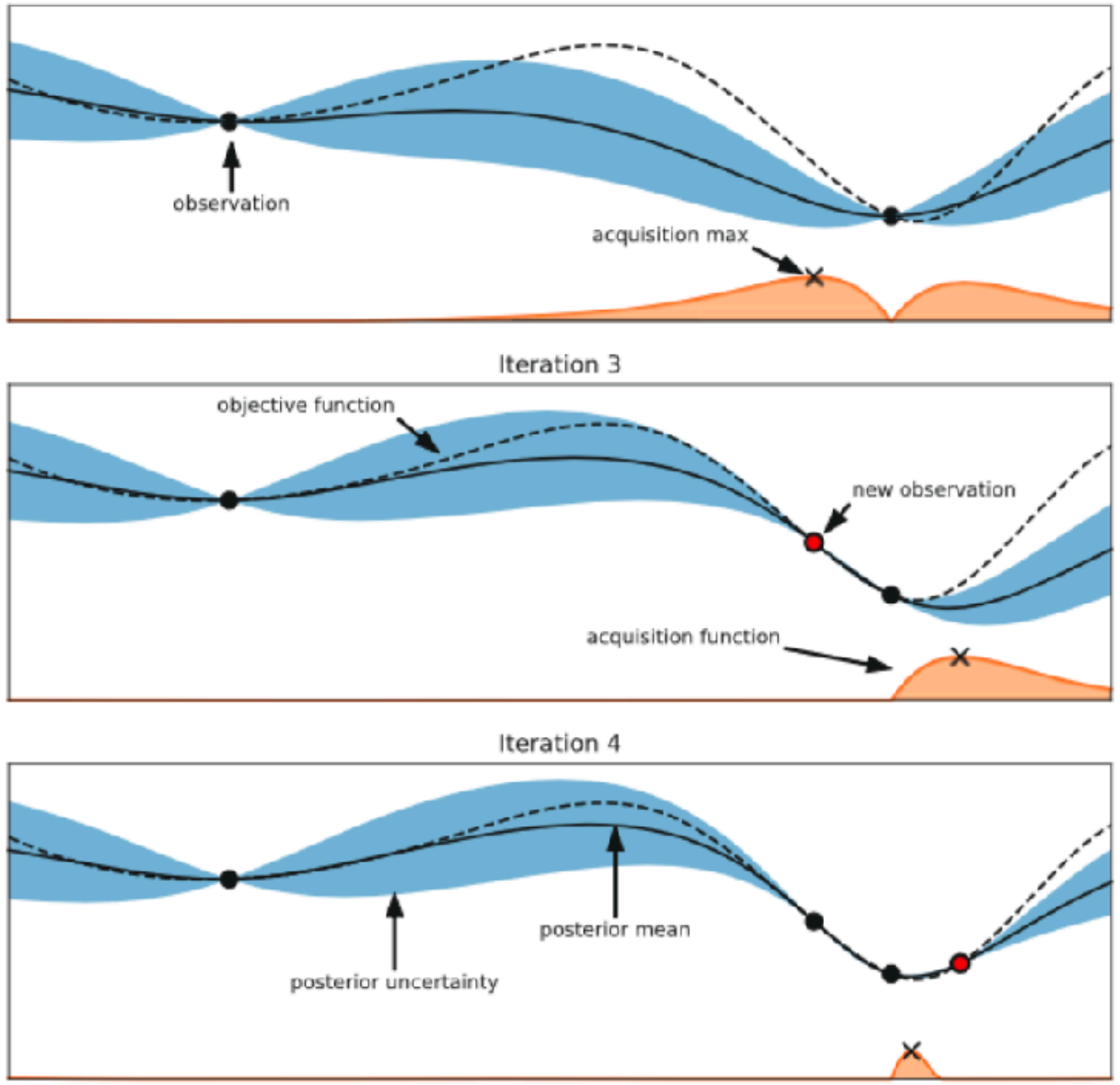
STRENGHTS

Dummy search

WEAKNESSES

Hyperparameters Optimization

BAYESIAN OPTIMIZATION



Gussian Process + Acquisition Function

Strong final performance

STRENGHTS

Computational costs

Not suitable for categorical HPs

WEAKNESSES

Hyperparameters Optimization

BOHB

BO

STRENGHTS

Strong final performance

WEAKNESSES

Computational costs

Not suitable for categorical HPs

HB

STRENGHTS

Strong anytime performance

Computational costs

WEAKNESSES

Dummy search

BO → **BOHB** ← HB

BO

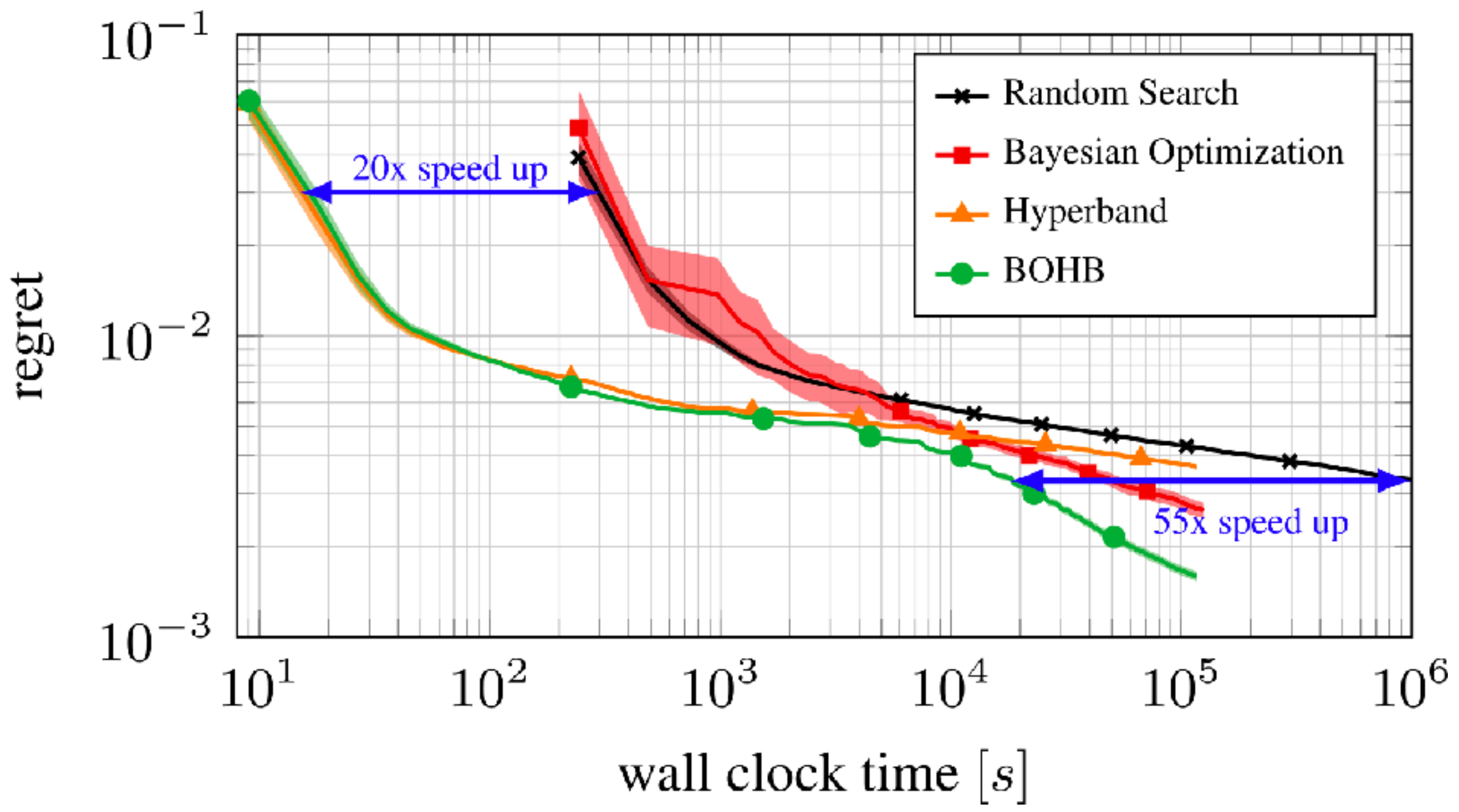
HB

BOHB

Relies on HB to determine how many configuration to evaluate with which budget

Uses TPE as BO component

Performs Successive Halving on model and random based picks



Damage Detector

NEED

Most online **insurance companies** can't verify the existence and **condition of the vehicle** during **policy application**, and this lead to fraudulent behaviours. For this reason, insurance companies limit the availability of online policies and require an insurance appraiser verification of the vehicle's conditions.

TASK

Develop a suite for **automated vehicle inspection** and offer **market-wide coverages**, currently available for already insured users only.

Damage Detector

WORKFLOW

Data Acquisition:
Web Scraping

Preprocessing:
Image Segmentation and
Data Augmentation

CNN hyperoptimization:
Random search *vs* BOHB

Transfer Learning hyperoptimization:
Random search *vs* BOHB

Results evaluation

Data Acquisition:

Web Scraping

Preprocessing:
Image Segmentation and Data Augmentation

CNN hyperoptimization:
Random search *vs* BOHB

Transfer Learning hyperoptimization:
Random search *vs* BOHB

Results evaluation

“Car model” + sinistrata

1643 Images

“Car model” + seconda mano

3604 Images

Data Acquisition:
Web Scraping

Preprocessing:

Image Segmentation and Data Augmentation

CNN hyperoptimization:
Random search *vs* BOHB

Transfer Learning hyperoptimization:
Random search *vs* BOHB

Results evaluation

Image Segmentation with ***Mask R-CNN***

Data Augmentation

CRIF DamageDetector

Data Acquisition:
Web Scraping

Preprocessing:

Image Segmentation and Data Augmentation

CNN hyperoptimization:
Random search *vs* BOHB

Transfer Learning hyperoptimization:
Random search *vs* BOHB

Results evaluation

Damaged cars:
40%

Not damaged cars:
60%

3015 images

Training set:
70%

Validation set:
15%

Test set:
15%

Dataset split

DamageDetector

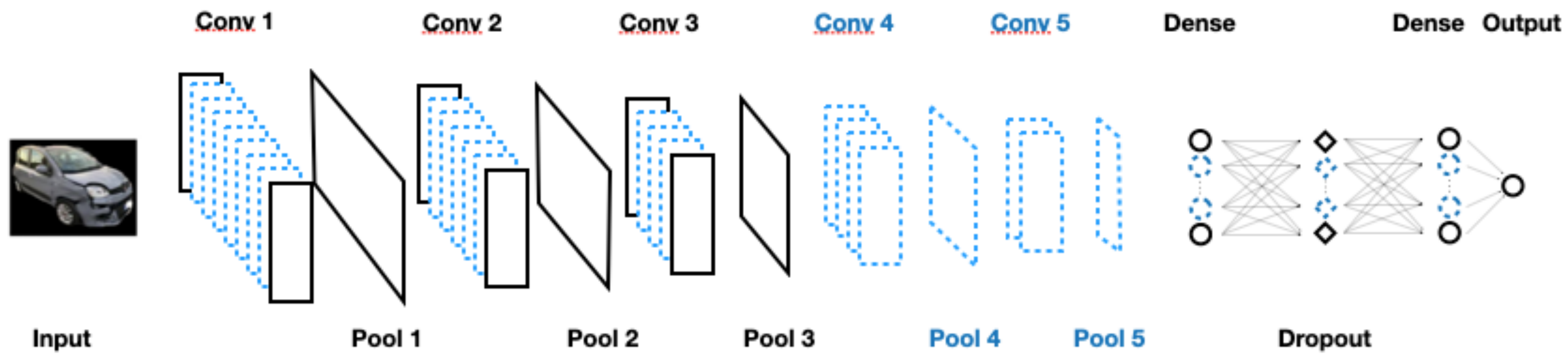
Data Acquisition:
Web Scraping

Preprocessing:
Image Segmentation and
Data Augmentation

CNN hyperoptimization:
Random search *vs* BOHB

Transfer Learning hyperoptimization:
Random search *vs* BOHB

Results evaluation



Hyperparameters	Range	Type
# Conv layers	[3,5]	Integer
# Neurons for each Conv layer	[4,64]	Integer
# Neurons for each Dense layer	[8,256]	Integer
Dropout rate	[0,0.9]	Float
Optimizer	{Adam, SGD}	Categorical
Learning rate	[1e-3, 1e-1]	Float
Momentum	[0,0.99]	Float
Epsilon	[1e-7, 1e-1]	Float

DamageDetector

Data Acquisition:
Web Scraping

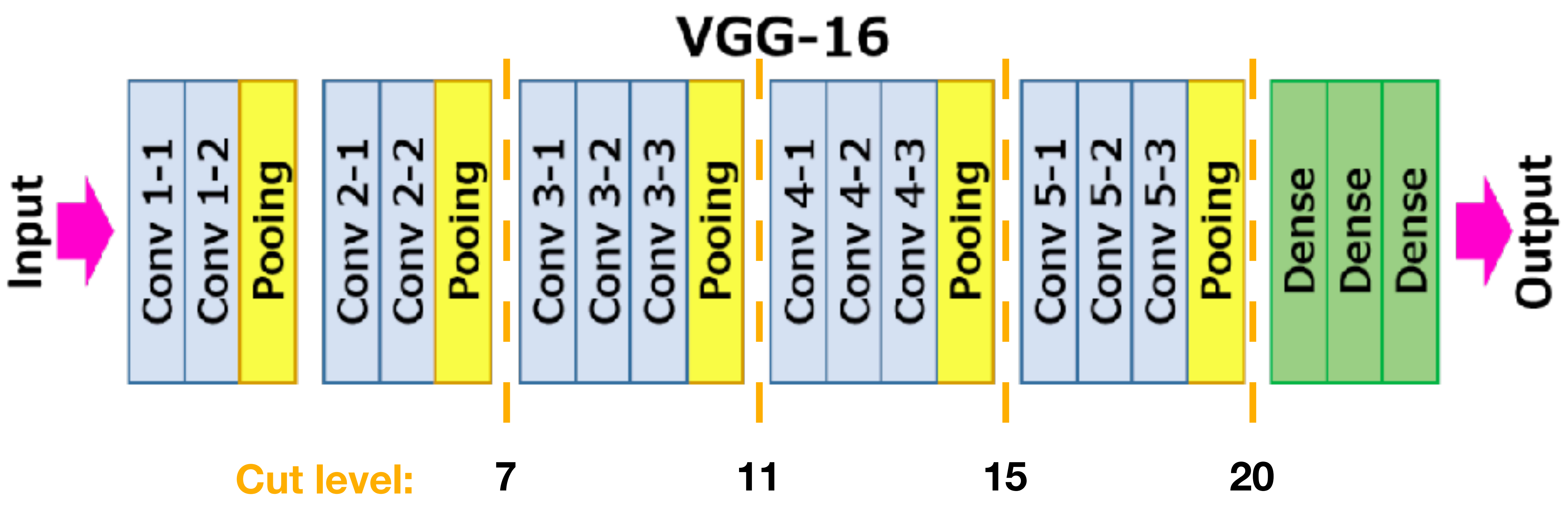
Preprocessing:
Image Segmentation and
Data Augmentation

CNN hyperoptimization:
Random search *vs* BOHB

Transfer Learning hyperoptimization:

Random search *vs* BOHB

Results evaluation



Hyperparameters	Range	Type
# Dense layers	[1,2]	Integer
# Neurons for each Dense layer	[8,256]	Integer
Dropout rate for each dr. layer	[0,0.5]	Float
Optimizer	{Adam, SGD}	Categorical
Learning rate	[1e-5, 1e-1]	Float
Momentum	[0,0.99]	Float
Epsilon	[1e-7, 1e-1]	Float
Cut level	{7,11,15,20}	Categorical

Data Acquisition:
Web Scraping

Preprocessing:
Image Segmentation and
Data Augmentation

CNN hyperoptimization:
Random search *vs* BOHB

Transfer Learning hyperoptimization:
Random search *vs* BOHB

Results evaluation

Table 3.10: Results comparison on validation set

Experiment	Accuracy	Precision	Recall	F1-measure
Random Search on CNN	0.701	0.740	0.790	0.760
BOHB on CNN	0.718	0.680	0.890	0.770
Random Search on TL	0.854	0.830	0.960	0.890
BOHB on TL	0.867	0.890	0.890	0.890

Table 3.11: Results comparison on test set

Experiment	Accuracy	Precision	Recall	F1-measure
Random Search on CNN	0.685	0.740	0.820	0.780
BOHB on CNN	0.719	0.680	0.920	0.780
Random Search on TL	0.846	0.820	0.930	0.870
BOHB on TL	0.888	0.890	0.910	0.900

DamageDetector

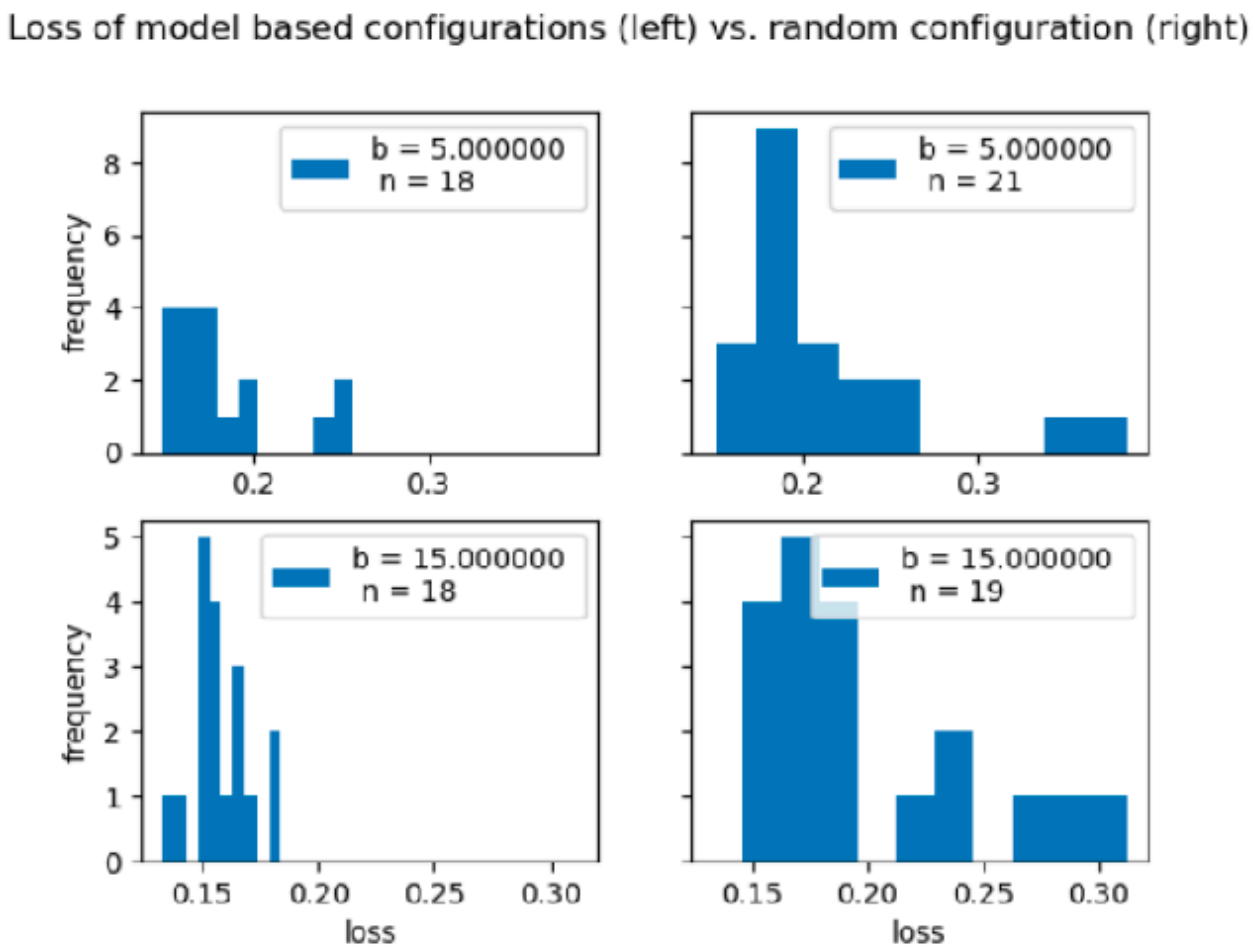
Data Acquisition:
Web Scraping

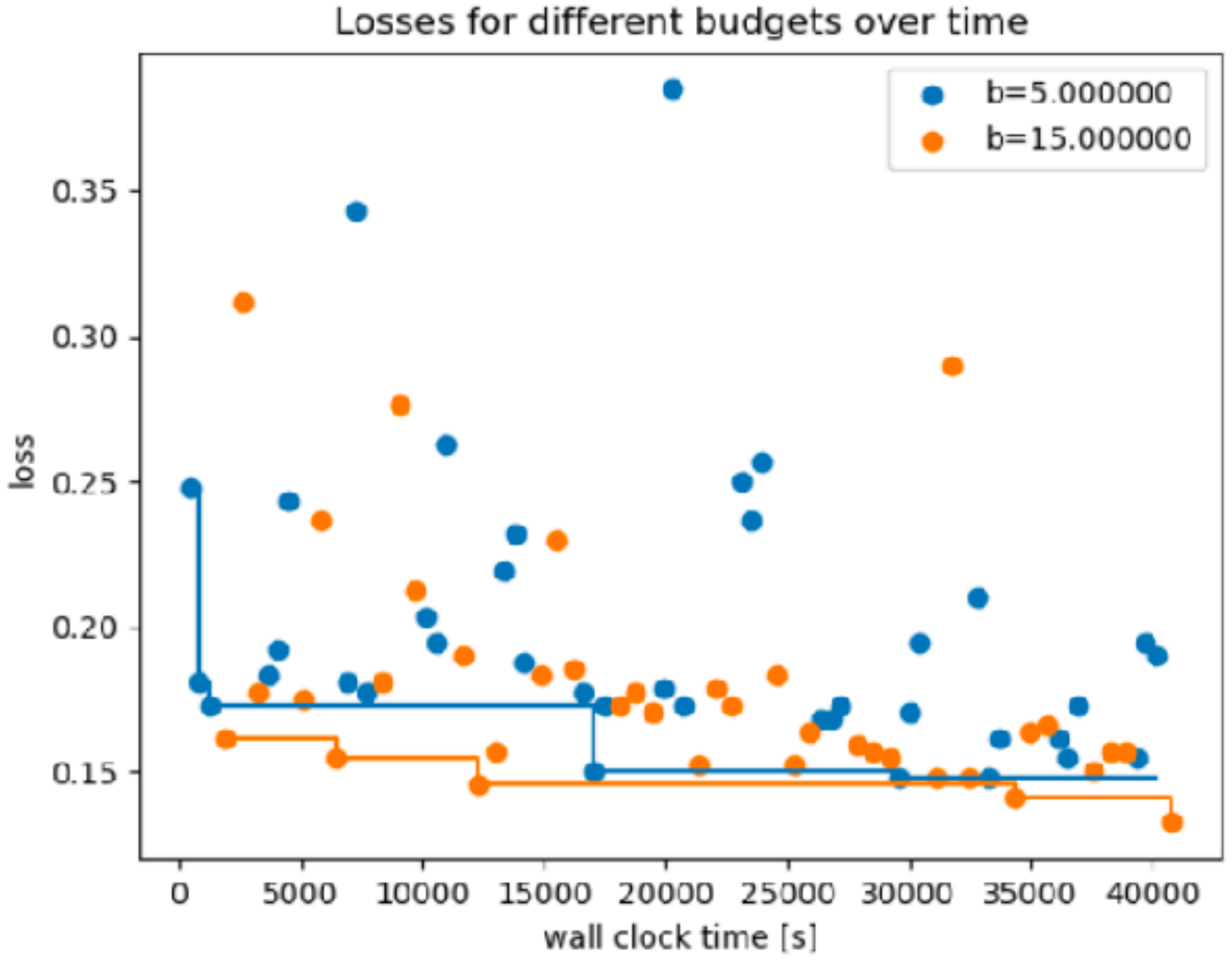
Preprocessing:
Image Segmentation and Data Augmentation

CNN hyperoptimization:
Random search *vs* BOHB

Transfer Learning hyperoptimization:
Random search *vs* BOHB

Results evaluation





DamageDetector

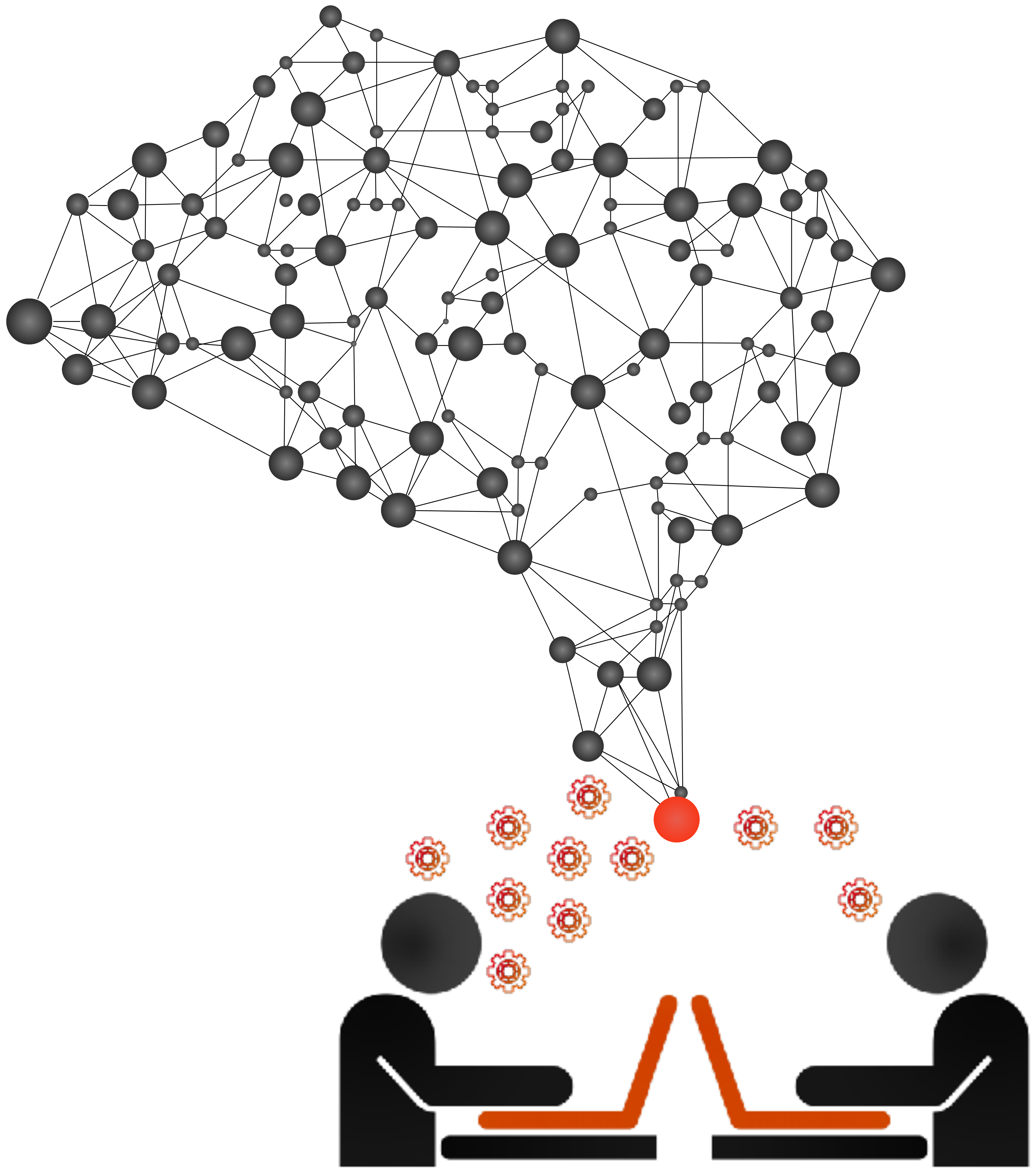
Conclusions

… and Next Steps

Input data volume matters!

BOHB performs better when applied to performing workflows.

Some adjustments can be made on bracket dimensions during the iterations.



Thanks