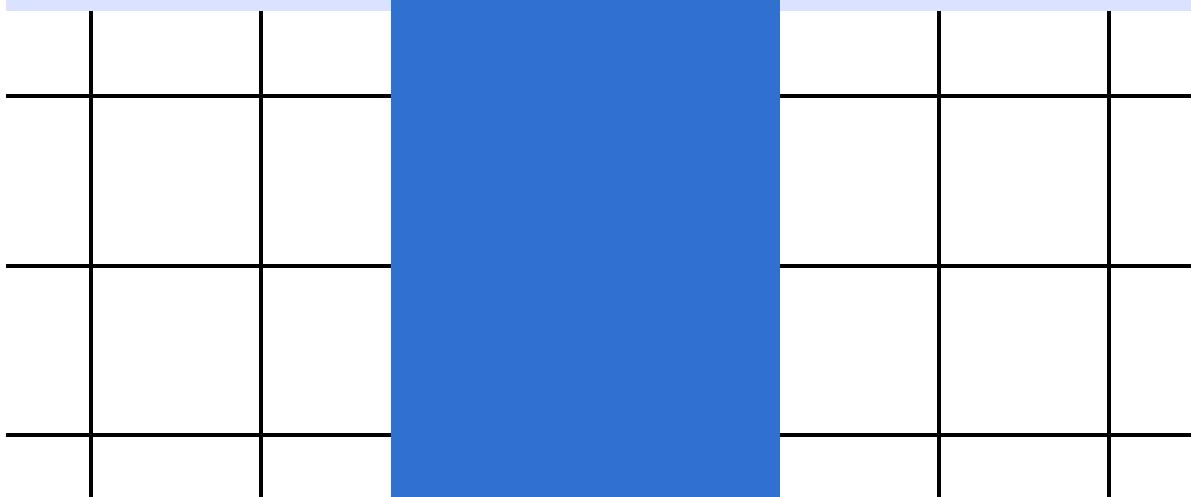


DATA AUGMENTATION AND PERFORMANCE

– Matee Vadrakchid –

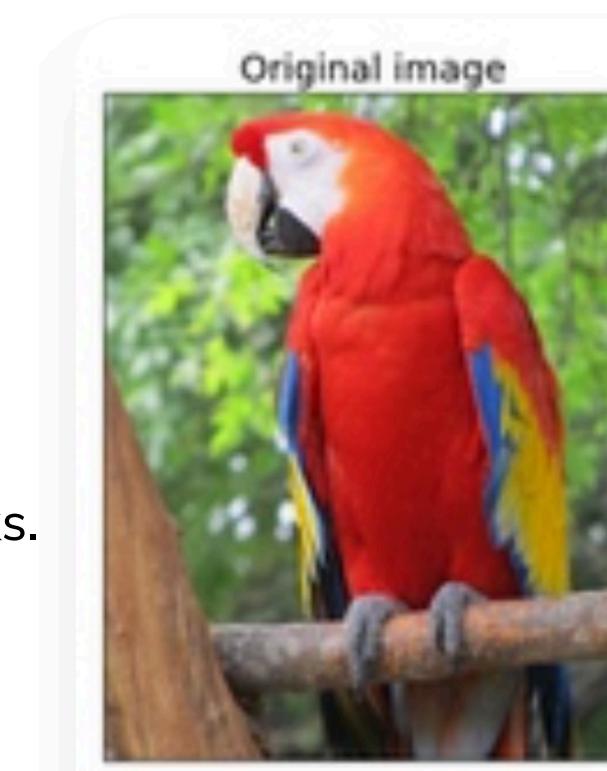


Part 1: Data Augmentation

Part 1: Data Augmentation

Do more with less data ?

Albumentations is a computer vision tool that boosts the performance of deep convolutional neural networks.



Part 1: Data Augmentation

Link: https://albumentations.ai/docs/getting_started/bounding_boxes_augmentation/

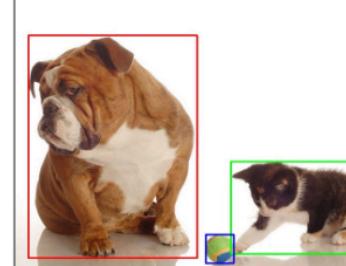
The diagram illustrates the Albumentations data augmentation pipeline. It starts with an input image showing a dog and a cat with three different colored bounding boxes (red, green, blue). A green arrow points from this image to a Python code block. The code defines a transformation function that takes an image and bounding boxes as input and returns an augmented image and augmented bounding boxes. The augmented image shows the same scene with the bounding boxes slightly altered. Another green arrow points from the augmented image to its corresponding JSON data representation, which lists the new bounding box coordinates and labels.

Next, you pass an image and bounding boxes for it to the `transform` function and receive the augmented image and bounding boxes.

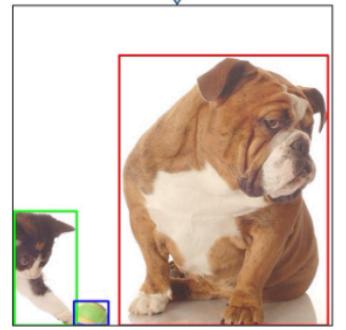
Python

```
1 | transformed = transform(image=image, bboxes=bboxes)
2 | transformed_image = transformed['image']
3 | transformed_bboxes = transformed['bboxes']
```

Input Image Bounding Boxes:



Augmented Image Bounding Boxes:



Original Bounding Boxes (JSON):

```
[  
    [23, 74, 295, 388, 'dog'],  
    [377, 294, 252, 161, 'cat'],  
    [333, 421, 49, 49, 'sports ball'],  
]
```

Transformed Bounding Boxes (JSON):

```
[  
    [149, 69, 295, 381, 'dog'],  
    [0, 289, 89, 161, 'cat'],  
    [85, 416, 49, 34, 'sports ball'],  
]
```

Part 2: Performance

Part 2: Performance

METRICS FOR MULTI-CLASS CLASSIFICATION: AN OVERVIEW

A WHITE PAPER

Margherita Grandini
CRIF S.p.A.*
m.grandini@crif.com

Enrico Bagli
CRIF S.p.A.*

Giorgio Visani
CRIF S.p.A.*
Department of Computer Science,[†]
University of Bologna

August 14, 2020

ABSTRACT

Classification tasks in machine learning involving more than two classes are known by the name of "multi-class classification". Performance indicators are very useful when the aim is to evaluate and compare different classification models or machine learning techniques. Many metrics come in handy to test the ability of a multi-class classifier. Those metrics turn out to be useful at different stage of the development process, e.g. comparing the performance of two different models or analysing the behaviour of the same model by tuning different parameters. In this white paper we review a list of the most promising multi-class metrics, we highlight their advantages and disadvantages and show their possible usages during the development of a classification model.

Link : <https://arxiv.org/pdf/2008.05756>

Part 2: Performance

Measuring Performance Terminology

- True Positive(TP): Data items that are correctly predicted
- True Negative(TN): Data items that are correctly predicted as a negative label
- False Positive(FP): Data items that are incorrectly predicted as a positive label
- False Negative(FN): Data items that are predicted as a negative label even though it is positive

Part 2: Performance

Measuring Performance

- Accuracy

$$\text{accuracy} = \frac{\# \text{correct}}{\# \text{total}}$$

- Precision

$$\text{precision} = \frac{TP}{TP + FP}$$

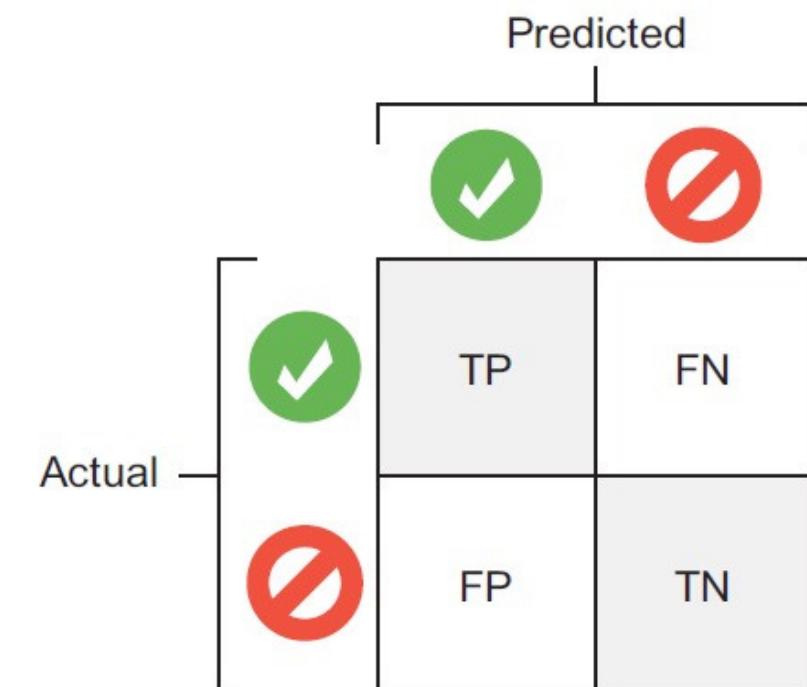
- Recall

$$\text{recall} = \frac{TP}{TP + FN}$$

Part 2: Performance

Confusion Matrix

- Confusion Matrix is a detailed report of machine learning performance



Part 2: Performance

Example for Cat Prediction

Confusion matrix		Predicted	
		Cat	Dog
Actual	Cat	30	20
	Dog	10	40

- Accuracy = 70/100
- Precision = 30/40
- Recall = 30/50

Part 2: Performance

If you have to make 100% recall for cat prediction, what will you do?

100% recall => don't miss any cat image

Always predict everything as a cat
=> it will create a lot of false positive

Part 2: Performance

Precision and recall

- Example :

	ω_1	ω_2	ω_3	ω_4
ω_1	97	0	2	1
ω_2	0	89	10	1
ω_3	0	0	100	0
ω_4	0	3	5	92

$$P = \frac{tp}{tp + fp}$$

$$R = \frac{tp}{tp + fn}$$

$$P_1 = 97/(97 + 0 + 0 + 0) = 100\%$$

$$P_2 = 89/(0 + 89 + 0 + 3) = 96.74\%$$

$$P_3 = 100/(2 + 10 + 100 + 5) = 85.47\%$$

$$P_4 = 92/(1 + 1 + 0 + 92) = 97.87\%$$

$$R_1 = 97/(97 + 0 + 2 + 1) = 97\%$$

$$R_2 = 89/(0 + 89 + 10 + 1) = 89\%$$

$$R_3 = 100/(0 + 0 + 100 + 0) = 100\%$$

$$R_4 = 92/(0 + 3 + 5 + 92) = 92\%$$

Part 2: Performance

Precision and recall

- F1: a more compact representation of the precision and recall properties of a system.

$$F1 = 2 \times \frac{precision \times recall}{precision + recall}$$

Part 2: Performance

Overfitting/Underfitting

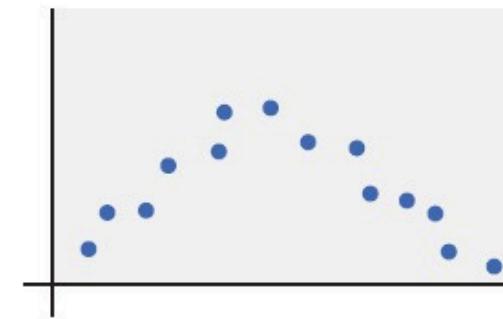
- With the machine learning, there are three scenarios that can happen

Train	Test	Result
Thumbs Up	Thumbs Up	Ideal
Thumbs Down	Thumbs Down	Underfit
Thumbs Up	Thumbs Down	Overfit

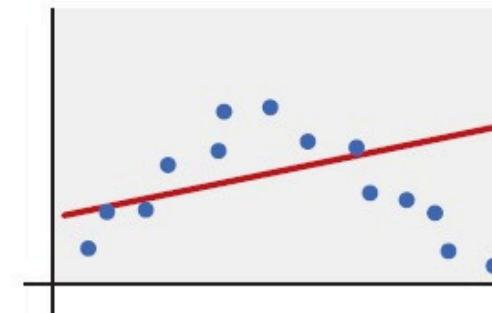
Part 2: Performance

Regression Model

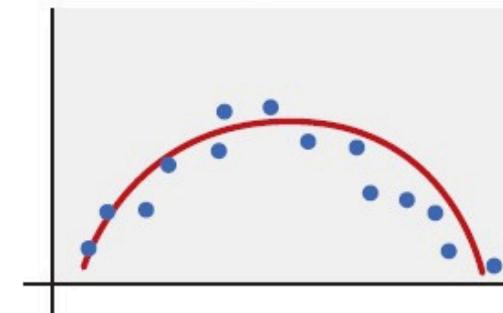
Raw data



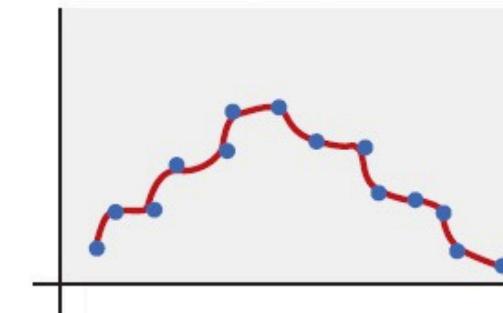
Underfit



Ideal fit



Overfit



Part 2: Performance

Ranked accuracy



Class Label	Probability
Airplane	0.0%
Automobile	0.0%
Bird	2.1%
Cat	0.03%
Deer	0.01%
Dog	0.56%
Frog	97.3%
Horse	0.0%
Ship	0.0%
Truck	0.0%

Class Label	Probability
Airplane	1.1%
Automobile	38.7%
Bird	0.0%
Cat	0.5%
Deer	0.0%
Dog	0.4%
Frog	0.11%
Horse	1.4%
Ship	2.39%
Truck	55.4%

Part 2: Performance

Rank-5 accuracy

- Step 1: Compute the class label probability for each input image
- Step 2: Sort the predicted class label probabilities in descending order with higher probability are placed at the front of the list
- Step 3: Determine if the ground-truth label exists in the top-5 predicted labels

Part 2: Performance

Rank-1 and Rank-5 Accuracy



Siberian husky



Eskimo dog

Two different classes in ImageNet from 1,000 classes

Part 2: Performance

Why Rank-5 Accuracy?

- With large classes, it is not easy to get high accuracy of rank-1
- Rank-5 can help see the performance even though we see less improvement in rank-1
- Some class are very similar

Assignment

You have a dataset of 155 images, each labeled as **Cat**, **Dog**, or **Person**. Your model tries to classify each image into one of these three classes. After prediction, you create this 3×3 confusion matrix:

	Predicted: Cat	Predicted: Dog	Predicted: Person	(Sum per true label)
True Cat	50	3	2	55
True Dog	2	45	3	50
True Person	4	6	40	50
(Sum per prediction)	56	54	45	155

Precision for one class (“Cat”)? Recall for one class (“Cat”)?

F1 Score for “Cat” ?