Gaetano Chiriaco 882638
Gianmarco Russo 887277

Pill Quality Control: Classification & GANs

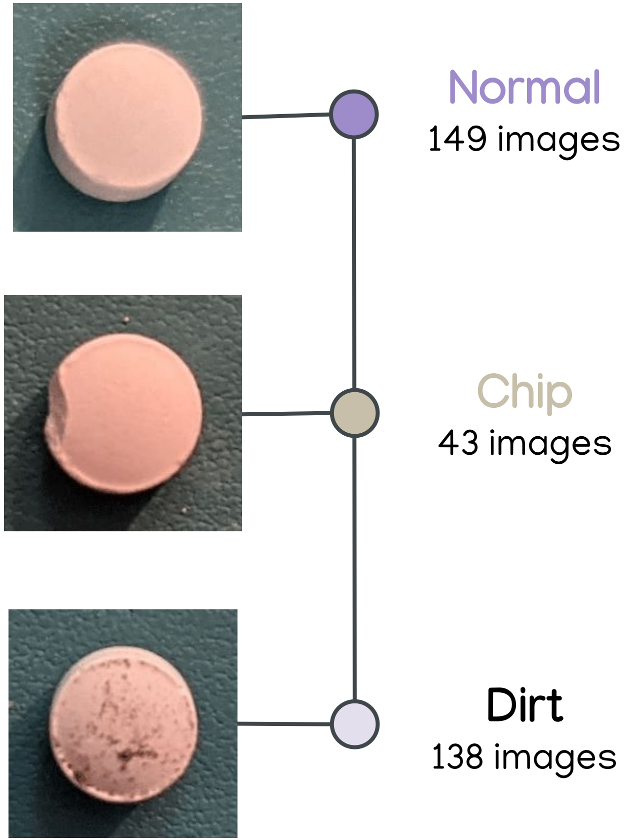
Problem Outline

Pill quality control dataset:

1. 225x225px RGB .jpg images
2. 3 different classes: *normal, chip, dirt*
3. Small dataset: **330** images
4. Unbalanced classes

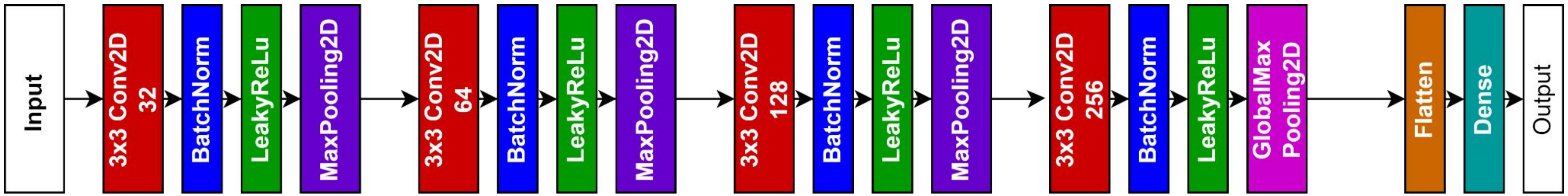
Chosen Tasks:

1. Image Classification
2. Image Augmentation using classic transformations and GANs



First Image classification

- *Stratified splitting in Train (60%), Validation (20%) and Test (20%)*
- *Convolutional Neural Network architecture:*



Callbacks: EarlyStopping (patience=5) *Loss: Categorical Cross-Entropy* *Optimizer: Adam (LR = 0.0001)*

$$\text{Loss} = -\sum_{i=1}^{\substack{\text{output}\\ \text{size}}} y_i \cdot \log \hat{y}_i$$

$$w_t = w_{t-1} - \eta \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon}$$

Other approaches used:

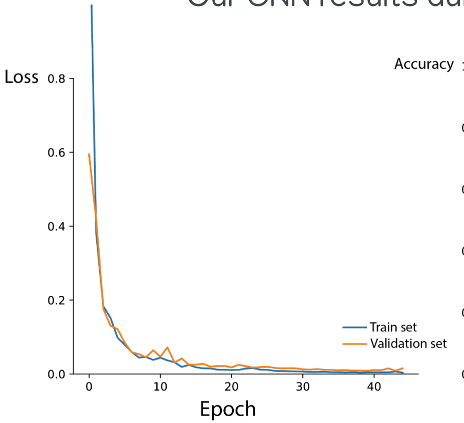
- **Data augmentation** using the aforementioned net (horizontal/vertical flip and random rotation)
- Fine tuning of an imageNet pre-trained network (**MobileNet**)

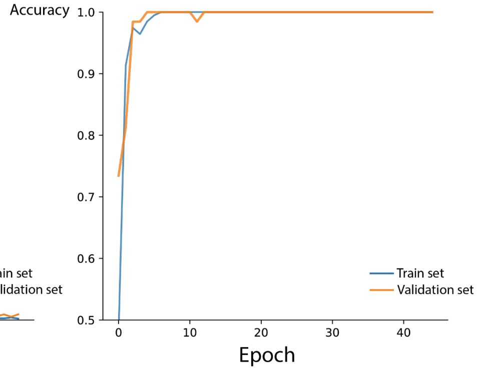
Results on the original dataset

Our CNN does a perfect job on the test set

	F-Score	Accuracy	Precision	Recall	Loss
Our CNN 390,083 trainable parameters	1	1	1	1	0.0193
MobileNet 1,865,283 trainable parameters	0.979	0.971	0.979	0.979	0.0359

Our CNN results during the training



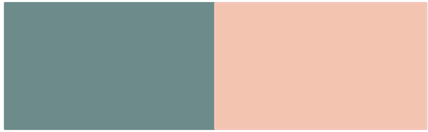


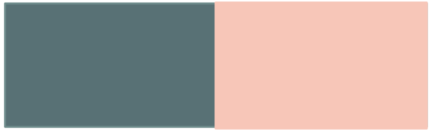
When things go too well there's almost always something wrong...

Fixing the biases - I

The models we trained have developed some biases: in particular the lighting and the background of the images is different among the classes




#6d8b8bff #f3c4b0ff


#587175ff #f7c6b8ff


#525753ff #e7a390ff

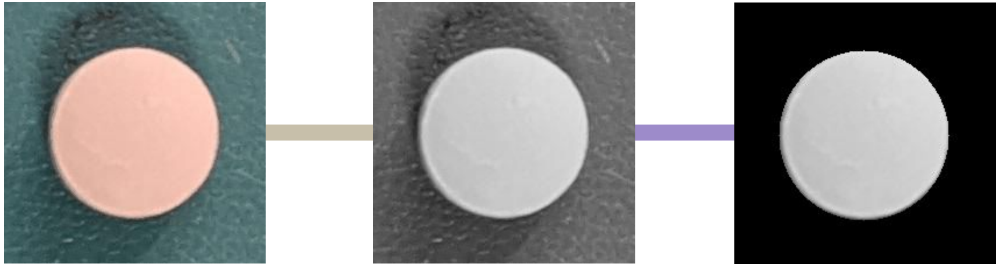
Probably the model learns to distinguish between classes focusing on other aspects instead of the pill itself

Fixing the biases - II

The training process has been repeated with **grayscale** and **masked** images.

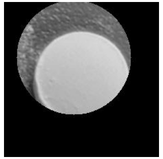
The objective was to eliminate the biases caused by the **background** and the **lighting.**

MobileNet was not usable at this point since it only works with **RGB** images, so we only used our CNN, slightly tuning its architecture to work with grayscale images.



RGB | GrayScale | Without background

Not all the images were perfectly masked, so we removed 40 images that were badly cut



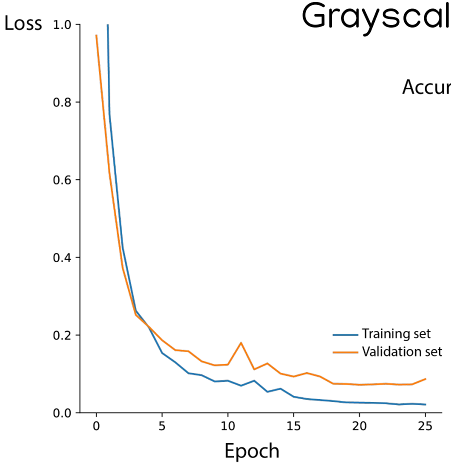
Python Library used: CV2

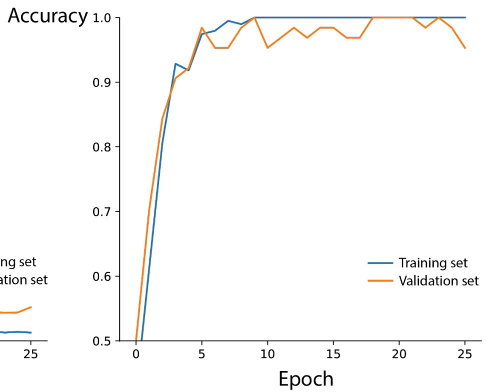
Results on altered images

	F-Score	Accuracy	Precision	Recall	Loss	Class Accuracy
CNN on grayscale images	0.989	0.9857	0.989	0.989	0.0568	Normal: 1 Dirt: 0.965 Chip: 1
CNN on masked images	0.769	0.787	0.835	0.718	0.5362	Normal: 0.9 Dirt: 0.79 Chip: 0.375

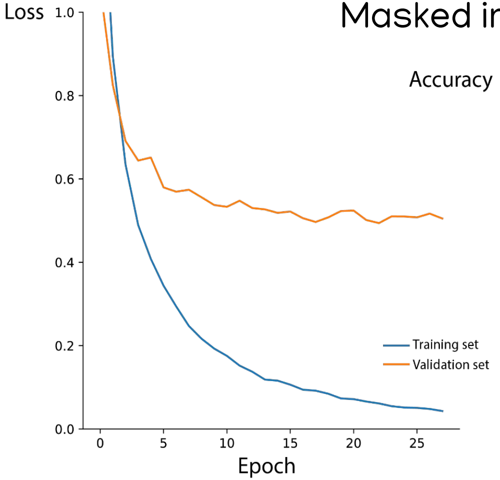
As we can see, the grayscale images don't fix the biases, showing the same performance. However the masked one definitely showed some signs of change: the model can't exploit the background and the lighting and its performance drops quite significantly.

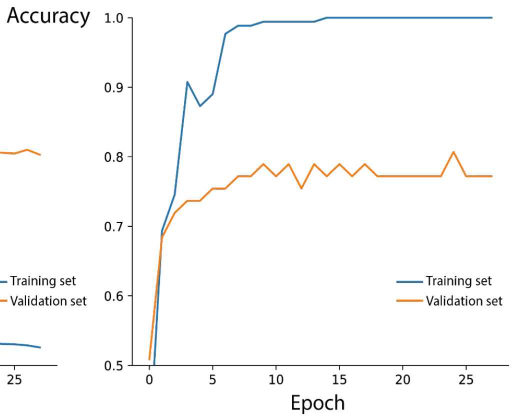
Grayscale images





Masked images



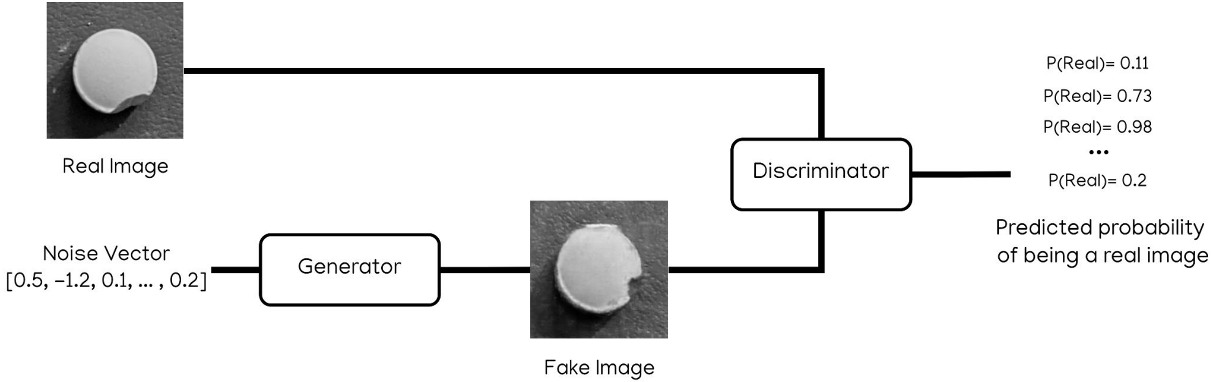


Generative Adversarial Neural Networks

GANs are generative models estimated via an **adversarial process,** in which we simultaneously train two models: a **generative model G** that captures the data distribution, and a d**iscriminative model D** that estimates the probability that a sample came from the training data rather than G.

Used for various tasks like:

- **Image augmentation**
- **Super resolution**
- **Style transfer**



Loss function: $$\min_G \max_D V(D,G) = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))]$$

GANs architecture for image augmentation

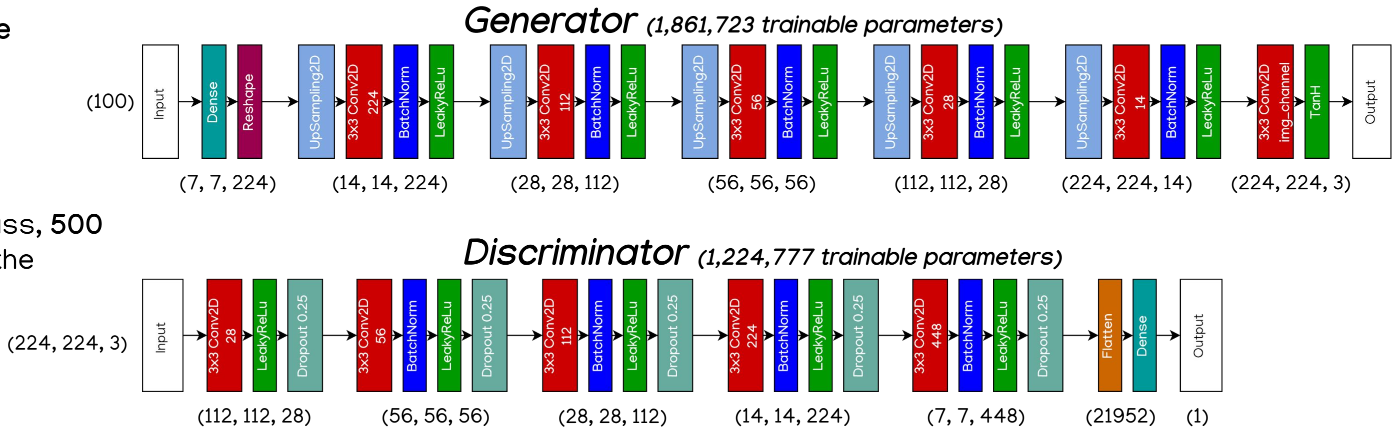
Different model for each **pill type** (normal, dirt, chip)

Different model for each **image type** (RGB, grayscale, masked)

500 epochs for the "normal" class, **500** for the "dirt" class and **500** for the "chip" class

Noise size: 100

Batch size: 16



Problems and disadvantages encountered:

- Complicated structure
- Needs usually a high number of epochs to output good (fake) images
- Mode Collapse
- Hard to evaluate the effectiveness

Data processing after GANs

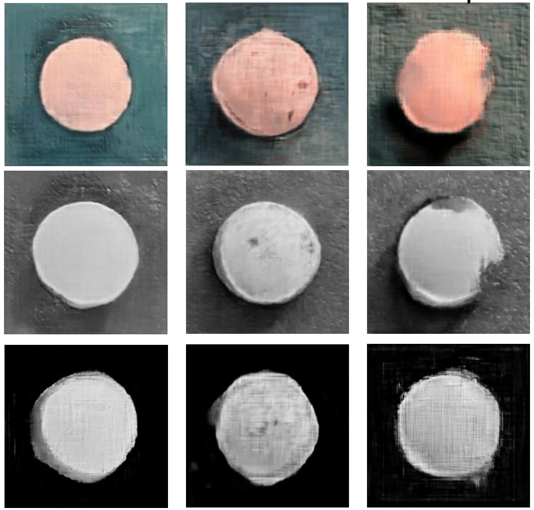
Last 200 epochs results

Normal | Dirt | Chip

RGB
(too many epochs to get a sufficient result)

Grayscale

Masked
(quality of generated images was poor)



Masking

Best GAN-generated images were <u>manually selected</u> and masked

Initial dataset
- Normal: 84 images
- Chip: 21 images
- Dirt: 68 images

+6 | +9 | +22

After GAN augmentation
- Normal (GAN augmented): 90 images
- Chip (GAN augmented): 30 images
- Dirt (GAN augmented): 90 images

+60

After classic augmentation
- Chip (GAN augmented): 90 images

Comparing the performances

	F-Score	Accuracy	Precision	Recall	Loss	Class Accuracy
CNN on augmented masked images	0.850	0.853	0.866	0.836	0.5326	**Normal:** 0.76 **Dirt:** 0.92 **Chip:** 1
CNN on masked images	0.769	0.787	0.835	0.718	0.5362	**Normal:** 0.9 **Dirt:** 0.79 **Chip:** 0.375

In this kind of problems the accuracy on the “anomalous” classes is really what matters

Possible developments and improvement

- Improve the masking process to avoid the removal of some of the images

- Explore different and more complicated types of GAN to obtain better results (like Conditional GANs)

- Solve the “Mode Collapse” issue

- Choose the best generation of images using more “formal” approaches (like Inception Score)

Gaetano Chiriaco 882638
Gianmarco Russo 887277

Pill Quality Control: Classification GANs

Thanks for your attention!

Bibliography:
1.GANs: [1406.2661] Generative Adversarial Networks (arxiv.org)

Sitography:
1.Transfer learning with MobileNet: https://www.tensorflow.org/tutorials/images/transfer_learning;
2.GAN | Generate Your Own Dataset using Generative Adversarial Networks (analyticsvidhya.com)