UNIVERSITY OF FERRARA
- EX LABORE FRUCTUS -

DE: Department of Engineering Ferrara

Laboratorio di Intelligenza Artificiale

Anomaly Detection on MVTec AD

Università degli Studi di Ferrara
Andrea Bazerla - 151792
Taoufik Souidi - 124485

Premessa

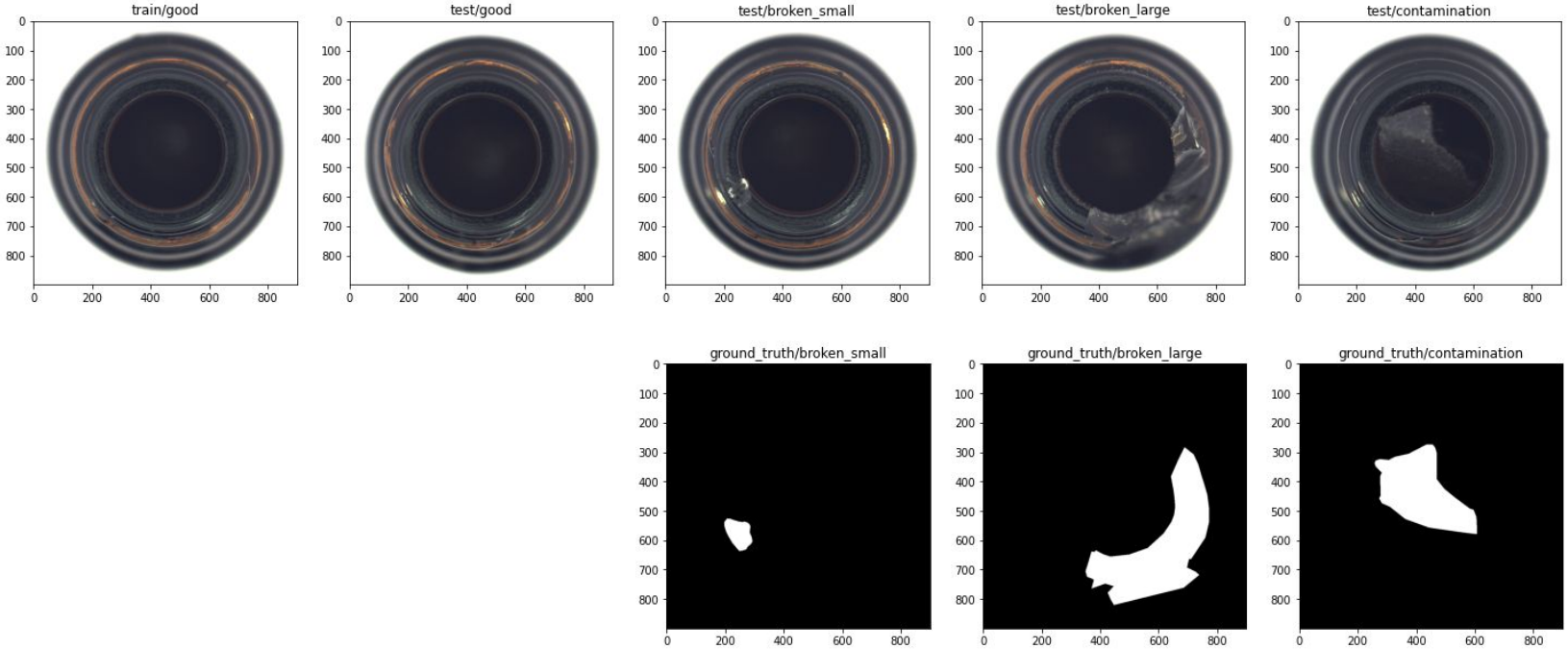
- **Problema**: gli esseri umani sono bravi a riconoscere difetti nelle immagini, ma l'ispezione manuale è dispendiosa, poco pratica e può portare ad errori.
- **Obiettivo**: analizzare una classe del dataset e riuscire a classificarne gli oggetti anomali da quelli non.
- **Soluzione**: sfruttare tecniche *unsupervised* di *Deep Learning* come quella del **Convolutional Autoencoder** basato su architettura *AlexNet*.

Dataset

- **Training Set**: immagini senza difetti (*Anomaly-Free*)
  Testing Set: immagini con e senza difetti
  Ground-Truth: annotazioni precise al pixel delle regioni anomale
- **15** classi di oggetti, **3629** immagini di training e **1725** testing
  Tipi di oggetti: 5 *textures* (Regolari e non) e 10 *oggetti* (Rigidi, deformabili e organici)
- **73** tipi di difetti, **5** per categoria
  Tipi di anomalie: *difetti superficiali* (Graffi, ammaccature), *anomalie strutturali* (Distorsioni), *parti di oggetti assenti*, ecc.
- **Risoluzione immagini**: 700x700~1024x1024, a colori e in scala di grigi

Scelte progettuali

- Texture difficili da analizzare per via dell’oscuramento delle immagini; le le anomalie tendevano ad essere rimosse.
- Oggetti senza simmetrie, non centrati e di diverse dimensioni difficili da ricostruire.
- Dataset ristretto alla classe ***“bottle”***: fondi di bottiglie di vetro circolari, centrati nelle immagini e tutti della stessa dimensione.

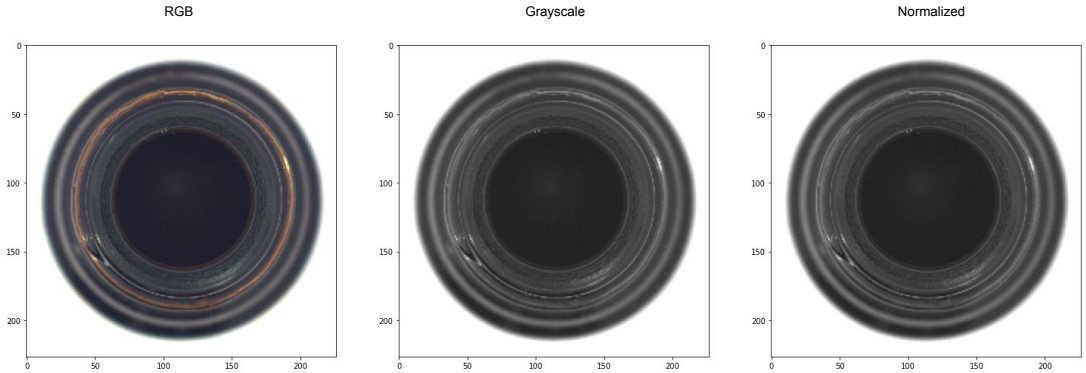

train/good
test/good
test/broken_small
test/broken_large
test/contamination
ground_truth/broken_small
ground_truth/broken_large
ground_truth/contamination

Data Augmentation

- **Problema**: training set composto SOLO da **209** immagini anomaly-free: non sufficienti per allenare il modello!
- **Soluzione**: **Data Augmentation**, mediante la duplicazione delle immagini e capovolgimento orizzontale e verticale per evitare l'overfitting del modello. Totale immagini = **627**

Pre-Processing

- Per migliorare le *performance* e *l'efficienza* della fase di training abbiamo pre-processato i dati.
- **Problema**: dato che le immagini in formato RGB hanno valori dei pixel [0, 255], l'addestramento avrebbe portato al vanishing/exploding del gradiente.
- **Soluzione**:
  - Conversione immagini RGB in scala di grigi (da 3 a 1 canali) per l'utilizzo della Structural Similarity Loss (SSIM)
  - Normalizzazione dei valori dei pixel [0, 1]
  - Riduzione dimensione immagini 227x227px (Immagine di input di AlexNet)


RGB
Grayscale
Normalized
0
50
100
150
200

Architettura

- **AlexNet** (2012, ILSVRC): restituisce la corretta etichetta tra 1000 classi su un dataset di più di un milione di immagini.
- **Architettura**:
  - 8 layer addestrabili: 5 convoluzionali e 3 fully-connected
  - ReLU come funzione di attivazione, eccetto per l'output layer con la softmax
- **Input**: immagini di dimensioni 227x227x3 px (RGB)
- **Numero totale parametri**: 60 664 758
- Possibilità di sfruttare 2 GPU in parallelo
- **Bottleneck**: dimensioni ridotte del layer fully-connected a 2048 neuroni.

Convolutional Autoencoder (1/2)

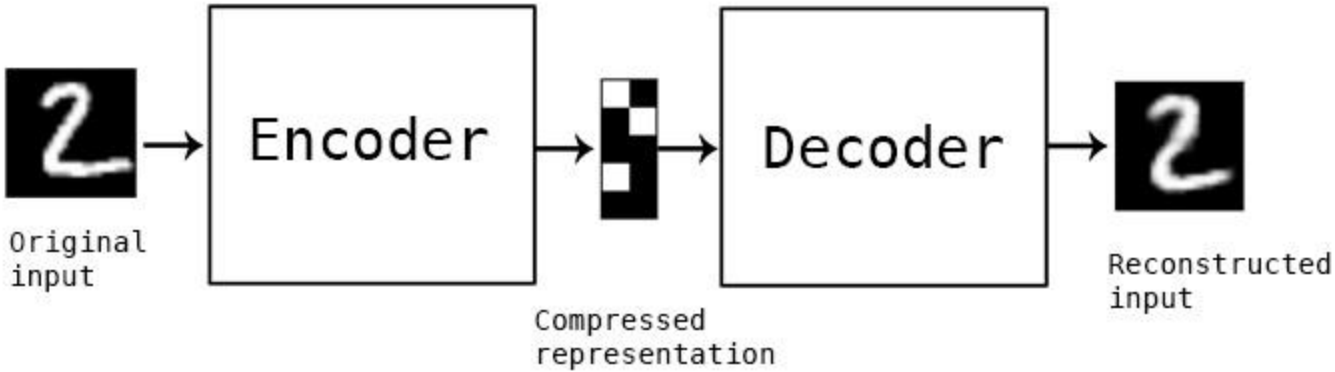
- **Convolutional Autoencoder**: rete neurale artificiale che modifica i suoi parametri addestrandosi nel ricostruire un'immagine dopo averne ridotto la dimensionalità.
- **Struttura**: è composto da un collo di bottiglia compreso tra **l'Encoder** che riduce l'immagine (Input) e il **Decoder** che ricostruisce l'immagine (Output). (Unsupervised Learning)

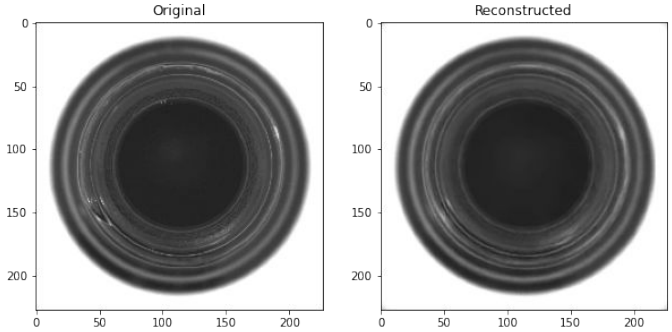
$$x' = D(\,E(\,x)\,)$$

$$x' \approx x$$

Convolutional Autoencoder (2/2)

- **Encoder**:
  - **Convolutional Layer**: sommatoria dei prodotti matriciali tra i kernel e l'immagine (Utile per il riconoscimento di linee, curve, bordi e patterns)
  - **Max-Pooling**: restituisce come risultato il massimo valore dell'area dell'immagine coperta dal kernel.
- **Decoder**:
  - **Convolutional Layer:** operazione inversa del Convolutional Layer
  - **Convolutional Transpose**: operazione inversa del Max-Pooling


Encoder
Decoder
Original input
Compressed representation
Reconstructed input

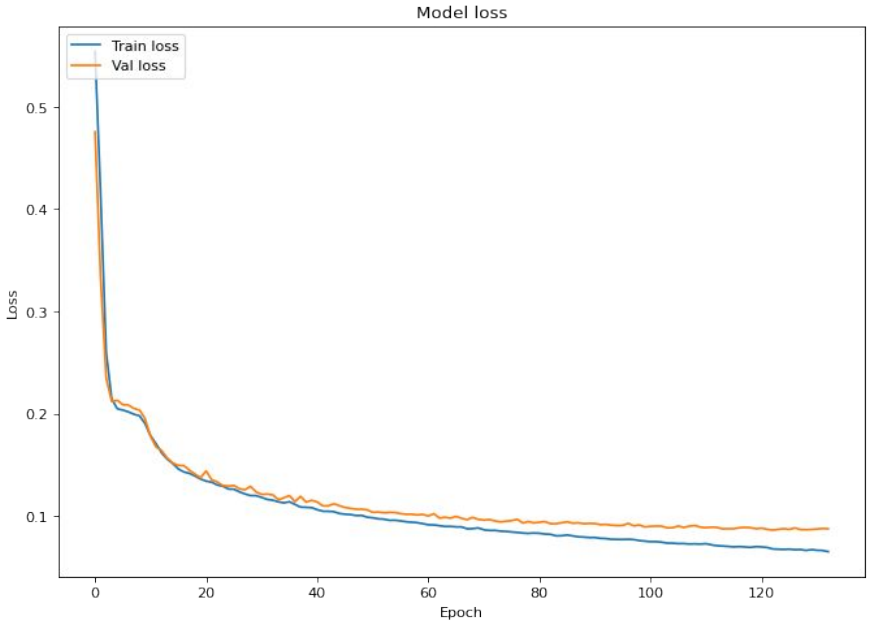

Original
Reconstructed
0
50
100
150
200

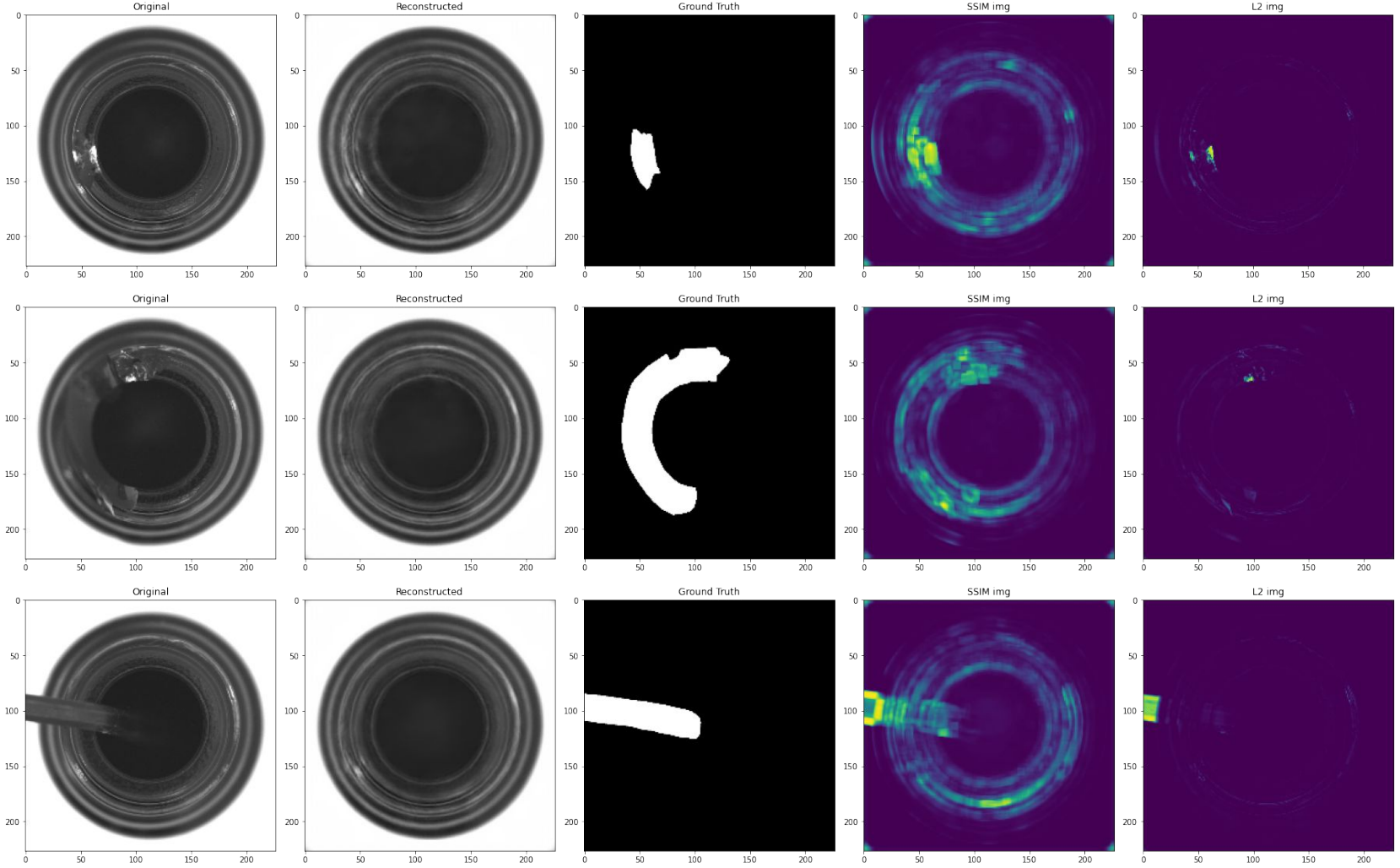
Structural Similarity Index (SSIM)

- Metrica utilizzata per misurare la somiglianza tra 2 immagini.
- Features estratte dalle immagini: **luminosità**, **contrasto** e **struttura**.
- Output: [-1, +1] se completamente differenti o uguali rispettivamente.
- Utilizzata come Loss function per il training e come metrica di valutazione per misurare l'errore tra le immagini di test e le rispettive ricostruzioni.

Training

- **Loss**: SSIM
- **Optimizer**: Adam
- **Learning Rate**: 1e-4
- **Batch Size**: 8
- **Epochs**: 150
- **Early Stopping**: 10
- **Validation Set**: 5%



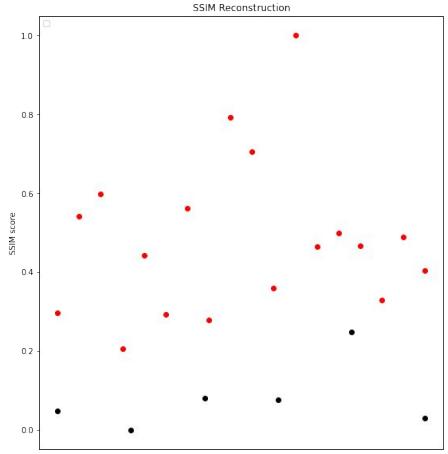

Original
Reconstructed
Ground Truth
SSIM img
L2 img
Original
Reconstructed
Ground Truth
SSIM img
L2 img
Original
Reconstructed
Ground Truth
SSIM img
L2 img

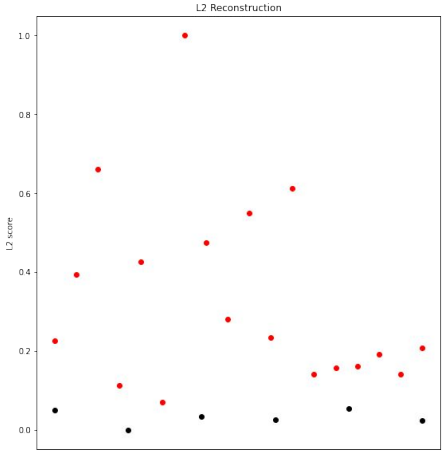
Valutazione

- Criteri di valutazione: ***qualitativo*** (Difference Map) e ***quantitativo*** (SSIM e L2).
- Classificazione oggetti anomali: **Difference Score**, media di tutti i valori dei pixel delle **Difference Map**

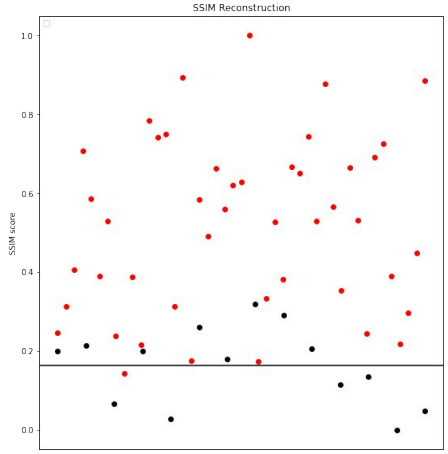
1. Alto errore di ricostruzione di un oggetto del Testing Set? **Oggetto anomalo!**
2. Quanto deve essere alto l'errore? **Trovare un *threshold*!**
3. Come trovare un threshold? Tramite un **Validation Set (30%)** e un **Test Set (70%)**
    - Grazie al ***Validation Set*** andiamo a cercare il threshold che divida gli oggetti anomali da quelli non con il minimo errore tra **falsi positivi** e **falsi negativi**.

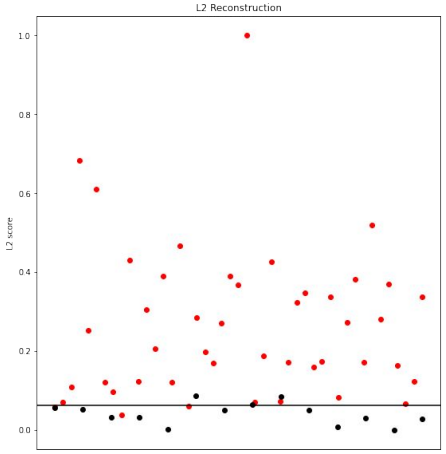
Difference Score Scatter of Validation images





Difference Score Scatter of Test images





F-Score

- **F-score**: misura dell'accuratezza sul Testing Set.
- Combina Precision e Recall insieme
- Valore migliore = 1

$$Precision = \frac{TN}{TN + FP}$$

$$Recall = \frac{TN}{TN + FN}$$

$$FScore = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall}$$

SSIM F-Score	0.57
L2 F-Score	**0.78**

Conclusioni

- SSIM più sensibile di L2, poiché penalizza anche la minima differenza
  -> SSIM buona per il training.
- La L2 classifica meglio rispetto alla SSIM gli oggetti anomali da quelli non
  -> L2 buona per la classificazione.