ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

DIPARTIMENTO DI INFORMATICA - SCIENZA E INGEGNERIA - DISI

**CORSO DI LAUREA MAGISTRALE IN
INGEGNERIA INFORMATICA**

VISUAL ANOMALY DETECTION: UN'ANALISI COMPARATIVA SULL'EVOLUZIONE DI METODI E DATASET PER APPLICAZIONI INDUSTRIALI

Tesi di laurea magistrale in Ingegneria Informatica

Relatore

Prof. Luigi Di Stefano

Correlatore

Alex Costanzino

Presentata da

Letizia Mancini

Sessione Marzo 2026

Anno Accademico 2024/2025

Abstract

La crescente automazione industriale ha reso il visual anomaly detection (il rilevamento automatico di difetti visivi) un problema centrale nel controllo qualità. La sfida di questo task è data dalla natura dei dati disponibili: i difetti sono rari e variabili nelle loro possibilità, portando ad avere solo campioni normali durante l'addestramento.

La tesi propone un'analisi comparativa dell'evoluzione di metodi e dataset per il visual anomaly detection industriale, dal 2017 al 2025. Dopo aver introdotto i fondamenti teorici del machine learning, delle reti neurali convoluzionali e del transfer learning, viene ripercorso cronologicamente lo sviluppo dei principali approcci di deep learning: dai metodi generativi basati su GAN e autoencoder, ai metodi feature-based con backbone pre-addestrati (PaDiM, SPADE, PatchCore), fino ai normalizing flow, alle tecniche di sintesi sintetica di anomalie (CutPaste, DRAEM), ai modelli student-teacher avanzati (EfficientAD, RD++) e ai vision-language model (WinCLIP, AnomalyCLIP). Viene inoltre analizzata l'anomaly detection su dati tridimensionali, con metodi feature-based per point cloud e framework multimodali RGB+3D.

Allo stesso tempo, viene tracciata l'evoluzione di quattordici benchmark standardizzati, da MVTec AD fino ai dataset più recenti che introducono anomalie logiche, dati 3D e condizioni operative reali.

Il panorama che emerge è quello di un campo ancora in crescita, in cui l'integrazione multimodale, la generalizzazione cross-domain e il passaggio da sistemi di semplice detection a maggiore comprensione rappresentano le direzioni di ricerca più rilevanti per il futuro.

Contents

Chapter 1

Introduzione

Nell'ultimo decennio, la crescente automazione dei processi produttivi ha trasformato radicalmente le aspettative nei confronti dei sistemi di controllo qualità. Individuare un difetto su un componente meccanico, riconoscere un'imperfezione su una superficie verniciata, rilevare un'anomalia in un circuito stampato: operazioni che un tempo richiedevano un operatore umano vengono oggi delegate, con risultati sempre più affidabili, a sistemi di visione artificiale basati su algoritmi di deep learning. Al centro di questa trasformazione si colloca il problema del *visual anomaly detection*, ovvero il rilevamento automatico di pattern visivi inattesi in immagini di prodotti industriali.

L'anomaly detection industriale si distingue dalla classificazione convenzionale per una caratteristica fondamentale: durante l'addestramento, il sistema dispone esclusivamente di esempi normali, poiché i difetti sono rari, costosi da raccogliere e, per natura, imprevedibili nella loro manifestazione. Questa situazione ha portato allo sviluppo di una famiglia eterogenea di approcci, ciascuno con assunzioni, punti di forza e limitazioni proprie: dai metodi basati su autoencoder e reti generative avversarie, fino ai più recenti modelli student-teacher, alle architetture memory-based e ai vision-language model.

In parallelo, sono stati definiti dataset standardizzati che si sono imposti come benchmark, rendendo possibile misurare con precisione i progressi del campo e indirizzare lo sviluppo verso le sfide ancora aperte.

L'obiettivo di questa tesi è offrire un'analisi comparativa dell'evoluzione dei metodi e dei dataset per il visual anomaly detection in ambito industriale, tracciando un percorso che parte dalle fondamenta teoriche dell'apprendimento automatico e arriva agli sviluppi più recenti del 2025. Il lavoro si propone non solo di catalogare i contributi della letteratura, ma di metterli in relazione tra loro, evidenziando le linee evolutive, le discontinuità paradigmatiche e le tendenze ancora in corso.

La trattazione è organizzata in tre capitoli.

Il **Capitolo 1** introduce i fondamenti teorici necessari alla comprensione del resto della tesi: si parte dalla definizione formale di machine learning e dei suoi

paradigmi principali (apprendimento supervisionato, non supervisionato e per rinforzo) per poi approfondire i concetti di representation learning, reti neurali feedforward e ricorrenti, algoritmi di training e tecniche di regolarizzazione. Il capitolo si conclude con un'analisi delle reti neurali convoluzionali (CNN), esaminando i principi fondamentali (connettività locale, condivisione dei pesi e pooling), l'evoluzione storica delle principali architetture e il ruolo del transfer learning, tecnica che si rivelerà centrale in quasi tutti i metodi di anomaly detection discussi nei capitoli successivi.

Il **Capitolo 2** costituisce il nucleo della tesi. Dopo aver formalizzato il problema dell'anomaly detection e discusso le sue specificità nel contesto industriale, viene ripercorsa cronologicamente l'evoluzione dei metodi di deep learning per il rilevamento di anomalie visive. L'analisi prende le mosse dai primi approcci basati su reti generative avversarie (AnoGAN, GANomaly, f-AnoGAN) e dagli autoencoder deterministici e variazionali, per poi esaminare la svolta rappresentata dai metodi feature-based (PaDiM, SPADE, Uninformed Students), i modelli a normalizing flow (CFLOW-AD, FastFlow, CS-Flow), le tecniche di generazione sintetica di anomalie (CutPaste, DRAEM), i metodi memory-based (PatchCore, MemSeg), le architetture student-teacher avanzate (Reverse Distillation, RD++, EfficientAD) e i metodi semplificati come SimpleNet.

Il capitolo dedica infine una sezione ai vision-language model (WinCLIP, AnomalyCLIP) e all'anomaly detection su dati tridimensionali, con i metodi feature-based per point cloud (BTF, Shape-Guided, Reg3D-AD) e i framework multimodali RGB+3D (AST, M3DM, CFM, D^3M).

Ciascuna famiglia di metodi è analizzata nelle sue motivazioni, architettura, vantaggi e limitazioni, con particolare attenzione alle discontinuità che hanno definito il passaggio da un paradigma al successivo.

Il **Capitolo 3** sposta il focus dai metodi ai dati, esaminando l'evoluzione dei principali benchmark per l'anomaly detection industriale dal 2018 al 2025. Vengono analizzati in dettaglio quattordici dataset, mettendone in evidenza le scelte di design, le categorie di oggetti, le tipologie di difetti e le metriche di valutazione adottate.

L'analisi mostra come l'orizzonte delle sfide proposte si sia ampliato in maniera crescente: dalla detection di difetti superficiali su texture strutturate, fino a scenari che richiedono ragionamento logico e strutturale [1], acquisizioni 3D [2], modalità multiple [3], [4] e condizioni più vicine alla produzione reale [5], [6].

Nel complesso, il panorama che emerge è quello di un campo in rapida crescita, in cui metodi sempre più efficienti e generalizzabili si confrontano con benchmark sempre più ricchi e realistici. L'auspicio è che questa analisi possa costituire un punto di riferimento utile per chiunque si avvicini al visual anomaly detection industriale, facilitando l'orientamento in una letteratura densa e in continua evoluzione.

Chapter 2

Related Work

2.1 Machine Learning

Negli ultimi decenni, l'utilizzo di sistemi informatici per la risoluzione di problemi complessi ha subito una trasformazione radicale. Tradizionalmente, gli algoritmi richiedono la codifica esplicita di regole e procedure. Il Machine Learning (ML), al contrario, introduce un paradigma alternativo.

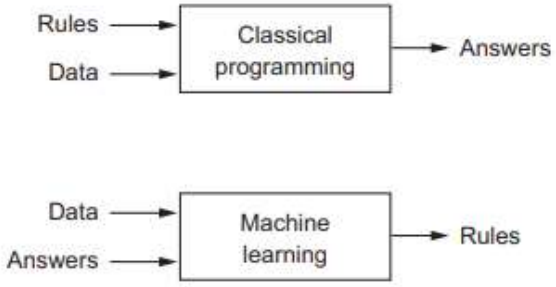

Figure 2.1: Confronto tra programmazione classica e machine learning. Nella programmazione tradizionale, regole esplicite trasformano dati in risposte. Nel machine learning, il sistema apprende le regole dai dati e dalle risposte fornite come esempi. Adattato da [7].

Con Machine Learning, infatti, si intende la disciplina dell'intelligenza artificiale che si occupa dello sviluppo di algoritmi e modelli statistici in grado di migliorare le proprie prestazioni in un determinato compito attraverso l'esperienza. La definizione classica proposta da Mitchell [8] formalizza questo concetto:

> Si dice che un programma per computer apprende dall'esperienza E rispetto a una classe di compiti T e a una misura di prestazione P,

se le sue prestazioni nei compiti T, misurate da P, migliorano con l'esperienza E.

La definizione individua i tre elementi costitutivi di ogni problema di apprendimento automatico: il **compito** (T), la **misura di performance** (P) e la **fonte di esperienza** (E).

L'apprendimento automatico si contrappone all'approccio tradizionale, dove regole e logica sono codificate esplicitamente. Nel ML, al contrario, il sistema inferisce pattern e regolarità in maniera autonoma dai dati, costruendo modelli predittivi che possono essere applicati a nuove osservazioni grazie ad un **framework probabilistico** [9].

2.1.1 Teoria della probabilità

L'incertezza è un concetto chiave nel riconoscimento di pattern: i dati osservati possono contenere rumore, essere incompleti o ambigui. Due regole fondamentali di probabilità determinato il ragionamento probabilistico [9, p. 14]: la *regola della somma*, che permette il calcolo delle distribuzioni marginali, e la *regola del prodotto*, che consente di calcolare probabilità congiunte. Da queste deriva il *teorema di Bayes*, che svolge un ruolo centrale nel machine learning e costituisce il fondamento dell'inferenza probabilistica:

$$p(Y|X) = \frac{p(X|Y)\,p(Y)}{p(X)} \tag{2.1}$$

dove X e Y sono variabili casuali, $p(Y|X)$ è la distribuzione *a posteriori*, $p(X|Y)$ è la *verosimiglianza*, $p(Y)$ è la distribuzione *a priori*, e $p(X)$ è l'*evidenza* (o probabilità marginale dei dati).

Nelle applicazioni pratiche di machine learning, dove spesso si lavora con dati multidimensionali come immagini, la formulazione si specializza considerando $\mathcal{X}$ come l'insieme dei dati osservati e $\mathbf{Y}$ come il vettore dei parametri o variabili latenti:

$$p(\mathbf{Y}|\mathcal{X}) = \frac{p(\mathcal{X}|\mathbf{Y})\,p(\mathbf{Y})}{p(\mathcal{X})} \tag{2.2}$$

Le credenze iniziali sui parametri (la *prior* $p(\mathbf{Y})$) vengono aggiornate alla luce dei dati osservati (tramite la *likelihood* $p(\mathcal{X}|\mathbf{Y})$) per ottenere la distribuzione *a posteriori* $p(\mathbf{Y}|\mathcal{X})$, che rappresenta la conoscenza aggiornata dopo aver visto i dati.

Nell'ambito del machine learning, emergono due approcci principali per l'interpretazione dell'incertezza: l'approccio *frequentista* e quello *Bayesiano* [9, pp. 19-22]. L'approccio frequentista stima un singolo set ottimale di parametri attraverso la *massima verosimiglianza* (maximum likelihood). L'approccio Bayesiano, invece, considera i parametri come variabili casuali e, partendo da una distribuzione a priori, la aggiorna alla luce dei dati per ottenere una distribuzione a posteriori.

2.1.2 Paradigmi di Apprendimento

Le metodologie di machine learning si dividono in tre approcci, quello **supervisionato**, quello **non supervisionato** e l'**apprendimento per rinforzo** [10].

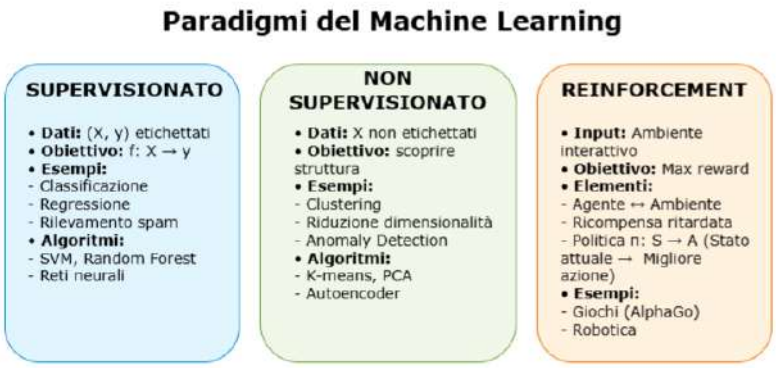

Figure 2.2: Schema dei tre paradigmi fondamentali del Machine Learning.

Apprendimento Supervisionato

L'apprendimento supervisionato rappresenta il paradigma più applicato [10]. L'obiettivo è apprendere una funzione di mapping che, dato un nuovo input $\mathbf{x}$, sia in grado di predire l'output $\mathbf{y}$ corrispondente con accuratezza.

In questo paradigma, il modello ha accesso a un dataset di addestramento, chiamato **training set** $\mathcal{D}$, composto da coppie input-output $\mathcal{D} = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^{N}$, con N esempi. dove $\mathbf{y}_i$ è l'etichetta o il valore target desiderato per l'input $\mathbf{x}_i$. L'obiettivo è apprendere una funzione di mappatura $f : \mathcal{X} \to \mathcal{Y}$ che generalizzi bene a nuovi input. Come evidenziato da [11], la tecnica si basa su una variabile dipendente (continua) per individuare il migliore fitting lineare con più variabili indipendenti.

L'apprendimento supervisionato si suddivide in due categorie principali in base alla natura dell'output:

Classificazione. Nei problemi di classificazione, il target appartiene a un insieme discreto di classi $y \in \{1, 2, ..., C\}$ [10, p. 3]. L'obiettivo è assegnare ciascun input a una delle C categorie disponibili. Esempi tipici includono il riconoscimento di cifre scritte a mano, dove l'input è un'immagine e l'output è una delle dieci cifre possibili, la classificazione di immagini o il rilevamento di email spam.

Gli algoritmi per la classificazione includono classificatori tradizionali come la regressione logistica, le Support Vector Machines (SVM), gli alberi decisionali, ma anche reti neurali e metodi ensemble come random forests e boosting [12]. Ciascun metodo presenta vantaggi e limitazioni in termini di interpretabilità, capacità di gestire dati ad alta dimensionalità, e robustezza al rumore.

Regressione. Nei problemi di regressione, l'output è una o più variabili continue $y \in \mathbb{R}$ [10, p. 3]. L'obiettivo è predire il valore di tali variabili dato un input, minimizzando tipicamente una funzione di loss come l'errore quadratico medio (Mean Square Error, MSE). Un esempio classico è la previsione del prezzo di una casa in funzione delle sue caratteristiche o le variazioni dei prezzi di mercato.

Tra gli algoritmi più utilizzati si annoverano la regressione lineare, la regressione ridge e lasso (per gestire la regolarizzazione, Shrinkage Methods), le reti neurali e metodi ensemble come random forests [12].

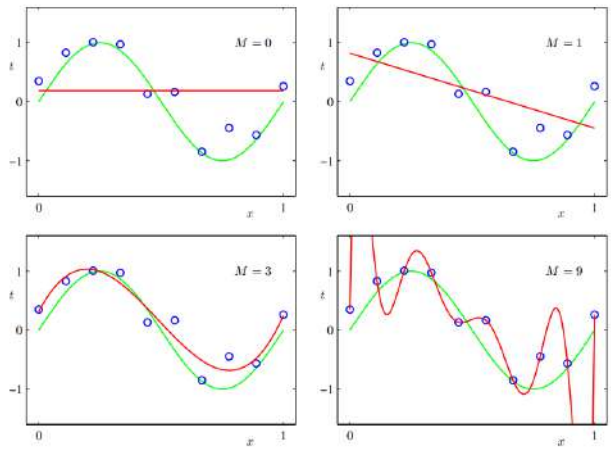

Figure 2.3: Illustrazione del fenomeno di overfitting: modelli di complessità crescente (polinomi di ordine M) applicati allo stesso dataset. Un modello troppo complesso ($M = 9$) si adatta al rumore nei dati piuttosto che alla funzione sottostante. Adattato da [9, Fig. 1.4].

Un problema fondamentale nell'apprendimento supervisionato è il fenomeno dell'*overfitting* [9, pp. 8-10]. Un modello eccessivamente complesso si adatta sia ai pattern nei dati, ma anche al rumore casuale presente nel training set. Questo compromette la capacità di generalizzare, determinando ottime prestazioni sui dati di addestramento conosciuti ma peggiori su dati nuovi. La scelta della complessità del modello richiede quindi un bilanciamento attento tra la capacità di catturare la struttura sottostante dei dati e il rischio di memorizzare il rumore.

Apprendimento Non Supervisionato

Nell'apprendimento non supervisionato vengono forniti solo dati non etichettati, senza risposte supervisionate. Formalmente, si considera un insieme di N osservazioni $(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N)$ di un vettore aleatorio $\mathbf{X}$ con densità congiunta $\Pr(\mathbf{X})$. L'obiettivo è inferire direttamente le proprietà di questa distribuzione di probabilità senza l'aiuto di un supervisore che fornisca un feedback [12].

Dal punto di vista probabilistico, questo si traduce nel compito di stima della **densità non condizionata** [10]. A differenza dell'apprendimento supervisionato, che mira a modellare la distribuzione condizionata $p(\mathbf{y}_i, \mid \mathbf{x}_i, \boldsymbol{\theta})$, l'apprendimento non supervisionato definisce modelli dalla forma $p(\mathbf{x}_i \mid \boldsymbol{\theta})$.

Gli algoritmi non supervisionati trovano applicazione in diversi contesti:

Clustering. Il clustering divide i dati in gruppi (o *clusters*) in modo tale che elementi all'interno dello stesso cluster siano più simili tra loro rispetto a elementi appartenenti a cluster diversi [9, p. 424]. Formalmente, data una collezione di osservazioni $\{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$, l'obiettivo è assegnare a ciascuna un'etichetta di cluster $z_i \in \{1, \ldots, K\}$ che rifletta la struttura dei dati [10].

Algoritmi classici includono: **metodi partitivi**, come K-means che minimizza la varianza intra-cluster, **metodi gerarchici** che costruiscono una gerarchia di raggruppamenti (agglomerativi o divisivi) [12], **model based clustering**, che modella i dati come provenienti da una miscela di distribuzioni probabilistiche, consentendo una stima probabilistica dell'appartenenza ai cluster [10]. Le applicazioni spaziano dall'identificazione di sottopopolazioni cellulari in biologia [13] alla segmentazione della clientela nell'e-commerce [14], fino alla scoperta di nuove classi di oggetti celesti in astronomia [15].

Dimensionality Reduction. La riduzione della dimensionalità ha come obiettivo quello di mappare dati ad alta dimensionalità in spazi di dimensioni inferiori, preservando le caratteristiche essenziali della distribuzione originale [9, pp. 559-577].

Questo processo consente di affrontare la "maledizione della dimensionalità" (*curse of dimensionality*), fenomeno per cui la densità dei dati decresce all'aumentare della dimensionalità, rendendo difficile l'apprendimento statistico [12].

I metodi principali si dividono in **lineari** e **non lineari**. Tra i primi, l'Analisi delle Componenti Principali (Principal Component Analysis, PCA) è quello più noto: proietta i dati lungo le direzioni ortogonali di massima varianza [16], permettendo una rappresentazione compatta che cattura la maggior parte dell'informazione.

Metodi non lineari comprendono gli embedding basati su vicinanze (*neighborhood based embeddings* come t-SNE (t-distributed Stochastic Neighbor Embedding) [17] e UMAP (Uniform Manifold Approximation and Projection) [18]: preservano relazioni locali tra punti, risultando efficaci per la visualizzazione di dataset complessi. Un'alternativa tipica del deep learning sono gli autoencoder [10], reti neurali che ricostruiscono il proprio input attraverso un collo di bottiglia (bottleneck layer) che codifica una rappresentazione compressa e informativa [19].

Tali metodi sono particolarmente efficaci per l'analisi di dati ad alta dimensionalità: dall'elaborazione di immagini [20], all'analisi di testi [21], fino alla preparazione di dati per task di machine learning, dove feature di dimensionalità ridotta possono migliorare efficienza computazionale e performance dei modelli [12].

Apprendimento per Rinforzo

L'apprendimento per rinforzo (reinforcement learning, RL) costituisce un terzo paradigma del machine learning, distinto sia dall'apprendimento supervisionato che da quello non supervisionato [22].

In questo scenario, un *agente* interagisce con un *ambiente* dinamico attraverso una sequenza di azioni: ad ogni passo, l'agente seleziona un'azione sulla base dello stato corrente dell'ambiente. Questo determina la transizione verso un nuovo stato e la ricezione di un segnale di *reward* retroattivo, che codifica il successo del compito.

A differenza del *supervised learning*, dove l'algoritmo apprende da esempi etichettati che indicano esplicitamente l'azione corretta e il compito del sistema è generalizzare le risposte, o dell'*unsupervised learning*, dove si cercano strutture in dati statici, nel **reinforcement learning** l'agente scopre autonomamente quali azioni producono le ricompense più elevate attraverso una ricerca *trial-and-error* [22, pp. 1-6]. L'agente riceve un segnale di ricompensa (reward) che indica la qualità dell'azione intrapresa rispetto all'obiettivo finale.

Matematicamente, il problema viene formalizzato come un *Markov Decision Process* (MDP), caratterizzato da: **stati** ($S_t \in \mathcal{S}$), **azioni** ($A_t \in \mathcal{A}(s)$) e **ricompense** ($R_{t+1} \in \mathcal{R} \subset \mathbb{R}$). Ad ogni step t, l'agente riceve una rappresentazione dello stato e seleziona l'azione successiva, determinando una transizione allo stato successivo dopo aver ottenuto la ricompensa. La traiettoria risultante è quindi della forma:

$$S_0, A_0, R_1, S_1, A_1, R_2, S_2, A_2, R_3, \ldots \tag{2.3}$$

In un MDP finito, la dinamica è completamente specificata dalla probabilità di **transizione** $p(s' \mid s, a)$ e dalla **distribuzione delle ricompense** $p(r \mid s, a, s')$, spesso combinate in $p(s', r \mid s, a)$. L'obiettivo è apprendere una *policy* $\pi : \mathcal{S} \to \mathcal{A}$ che massimizzi il return atteso, definito come la somma delle ricompense future scontate.

L'integrazione del reinforcement learning con il deep learning ha dato origine al *deep reinforcement learning* (DRL) [23], [24], che ha raggiunto risultati notevoli in domini complessi come videogiochi [25], robotica [26], giochi da tavolo [27], guida autonoma [28]. Nonostante i successi, tuttavia, il DRL presenta ancora sfide in termini di efficienza campionaria e stabilità dell'apprendimento [29].

2.1.3 Limitazioni del Machine Learning Tradizionale

Nonostante i successi dei metodi di machine learning tradizionali descritti in numerosi domini applicativi, l'applicazione di algoritmi tradizionali, sia supervisionati (SVM, decision trees) che non supervisionati (K-means, PCA), a dati complessi nella loro forma grezza (come immagini, audio o testo) presenta limitazioni intrinseche legate alla modalità con cui le rappresentazioni dei dati vengono costruite [30], [31].

Feature Engineering

La sfida principale nel machine learning tradizionale risiede nella *feature engineering*: indipendentemente dal paradigma di apprendimento adottato, i dati devono essere trasformati in rappresentazioni strutturate e informative prima di essere elaborati [30], [32]. In una prima fase, si progetta manualmente un estrattore di feature (*feature extractor*) per trasformare i dati *raw* in una rappresentazione strutturata e informativa. Solo in seguito, l'algoritmo di apprendimento poteva operare su queste feature per svolgere il compito desiderato (classificazione, regressione, etc.) [19].

Nel riconoscimento di oggetti in immagini, ad esempio, sia approcci supervisionati che non supervisionati richiedono la progettazione manuale di feature extractors specifici per catturare proprietà rilevanti come bordi, texture, angoli. Tra questi possiamo ricordare SIFT (Scale-Invariant Feature Transform) [33], HOG (Histogram of Oriented Gradients) [34], o filtri di Gabor [35]. Un classificatore supervisionato come SVM opera su queste features per separare le classi, mentre un algoritmo di clustering non supervisionato come K-means le utilizza per raggruppare immagini simili [10]. In entrambi i casi, tuttavia, le features sono progettate manualmente e non adattate al compito specifico.

Questo processo presenta diversi svantaggi: features progettate per un compito specifico raramente generalizzano ad altri contesti. Inoltre, la progettazione manuale porta ad una inevitabile semplificazione che potrebbe non catturare la complessità delle variazioni dei dati naturali. Come evidenziato da [31], *"The success of machine learning algorithms generally depends on data representation"* e rappresentazioni inadeguate limitano le prestazioni raggiungibili.

Scalabilità e Separazione delle Fasi

Quando applicati direttamente a dati ad alta dimensionalità (come immagini rappresentate da migliaia di pixel) i metodi tradizionali richiedono quantità di dati crescenti esponenzialmente con la dimensionalità, come discusso nella Sezione 2.1.2 riguardo alla *curse of dimensionality*. Questo problema riguarda sia metodi supervisionati che non supervisionati, rendendo l'apprendimento intrattabile senza tecniche di riduzione della dimensionalità [12].

Tipicamente, i metodi tradizionali separano la feature extraction e l'apprendimento del modello in fasi separate e indipendenti: le features vengono prima estratte (manualmente o con algoritmi non supervisionati come PCA o K-means), e solo successivamente viene applicato l'algoritmo di apprendimento, supervisionato o non [30]. Questa separazione impedisce l'ottimizzazione del processo: le features non vengono adattate per ottimizzare la performance sul compito finale, ma rimangono fisse e generiche [31].

Queste limitazioni hanno motivato lo sviluppo di approcci in grado di **apprendere automaticamente rappresentazioni efficaci** direttamente dai dati grezzi, eliminando la necessità di feature engineering manuale. Tale transizione ha dato origine ai modelli di **deep learning**: estendendo i principi del machine learning attraverso reti neurali con molti strati (*deep*), aumenta la

capacità di apprendere gerarchie di feature via via più astratte e potenti [19].

2.2 Deep Learning

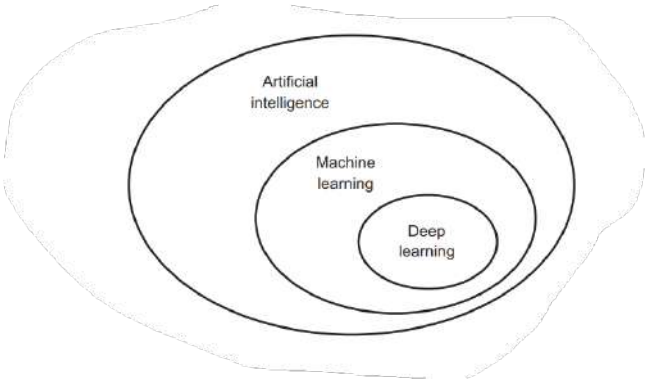

Figure 2.4: Relazione tra intelligenza artificiale, machine learning e deep learning. Adattato da [7].

Come evidenziato nella Sezione 2.1.3, le limitazioni del machine learning tradizionale hanno portato allo sviluppo di metodi capaci di apprendere automaticamente rappresentazioni dai dati grezzi. Il **Deep Learning** rappresenta una classe di tecniche di machine learning basate sull'idea di apprendere rappresentazioni efficaci direttamente dai dati stessi. L'obiettivo viene realizzato attraverso architetture composte da molteplici livelli di rappresentazione (reti neurali flessibili e multistrato, *deep*): tramite trasformazioni non lineari, le rappresentazioni diventano più astratte. Ogni livello della gerarchia costruisce rappresentazioni via via più astratte a partire da quelle del livello precedente. Con la composizione di un numero sufficiente di tali trasformazioni, possono essere apprese funzioni molto complesse.

2.2.1 Representation Learning

Il representation learning è un insieme di metodi che consentono l'apprendimento automatico delle rappresentazioni necessarie per il rilevamento o la classificazione a partire dai dati grezzi [31]. L'idea alla base è che una buona rappresentazione faciliti l'estrazione di informazioni utili ai fini di classificazione o previsione.

Inizialmente, la progettazione di sistemi di riconoscimento richiedeva la definizione manuale di feature extractor specifici per dominio, in grado di trasformare segnali grezzi in rappresentazioni strutturate adatte all'elaborazione [30]. Il representation learning, al contrario, apprende le informazioni direttamente dai dati: queste sono organizzate in maniera gerarchica, consentendo di rappresentare concetti complessi a partire da elementi più semplici [19].

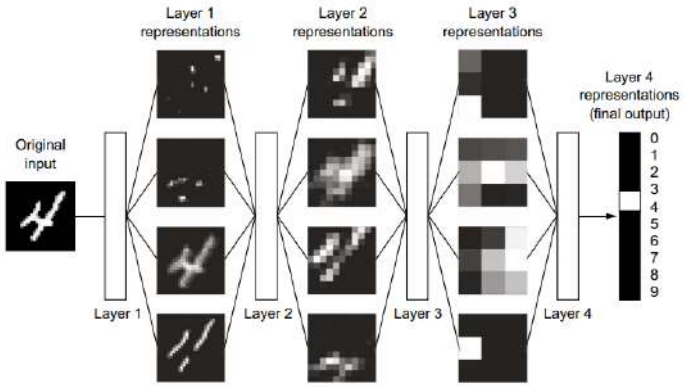

Figure 2.5: Esempio di apprendimento gerarchico di rappresentazioni in un modello di deep learning per la classificazione di cifre. Adattato da [7].

Il processo di apprendimento delle rappresentazioni avviene attraverso una **gerarchia di trasformazioni**. Come illustrato nella Figura 2.5, la rete trasforma l'immagine della cifra in rappresentazioni che diventano progressivamente più distanti dall'immagine iniziale ma più informative rispetto al risultato finale. Si può pensare ad una rete profonda come ad un processo di distillazione dell'informazione a più stadi, in cui i dati attraversano una serie successiva di filtri per emergere più utili al compito specifico [7].

2.2.2 Reti Neurali

Le reti neurali artificiali (Artificial Neural Networks, ANN) costituiscono il fondamento del deep learning. Sono chiamate *reti* perché sono tipicamente rappresentate componendo insieme molte funzioni diverse. Il modello è associato a un grafo diretto aciclico che descrive come le funzioni sono composte insieme. Ad esempio, potremmo avere tre funzioni $f^{(1)}$, $f^{(2)}$ e $f^{(3)}$ connesse in catena, per formare:

$$f(x) = f^{(3)}\big(f^{(2)}(f^{(1)}(x))\big)$$

Queste strutture a catena sono le più comuni nelle reti neurali. In questo caso, $f^{(1)}$ è il primo strato della rete, $f^{(2)}$ il secondo e così via. La lunghezza complesisva fornisce la profondità del modello, da cui il termine "deep learning" [19].

Infine, queste reti sono dette neurali perché sono liberamente ispirate alle neuroscienze. [36] offre una definizione precedente, ispirata all'analogia neurobiologica, descrivendo la rete come "un processore distribuito massivamente parallelo, composto da unità di elaborazione semplici, che possiede una propensione naturale ad immagazzinare conoscenza esperienziale e a renderla disponibile per l'uso."

Per comprendere come funzionano le reti neurali, è necessario partire dall'unità computazionale elementare: il percettrone (*perceptron*). Successivamente,

vedremo come la composizione di molteplici perceptron in architetture stratificate dia origine a sistemi capaci di apprendere funzioni arbitrariamente complesse.

Il Percettrone

Il perceptron [37] rappresenta il modello più semplice di neurone artificiale e costituisce l'unità computazionale fondamentale delle reti neurali. Ispirato da un modello semplificato del neurone biologico, il perceptron è un algoritmo di apprendimento supervisionato concepito per la classificazione binaria, implementando una trasformazione matematica che combina linearità e non linearità.

Formulazione matematica. Il perceptron corrisponde a un modello di classificazione a due classi in cui il vettore di input $\mathbf{x} = (x_1, x_2, \ldots, x_n) \in \mathbb{R}^n$ viene prima trasformato usando una trasformazione non lineare fissa per ottenere un vettore di feature $\boldsymbol{\phi}(\mathbf{x})$, che viene poi utilizzato per costruire un modello lineare generalizzato dalla forma [9]:

$$y(\mathbf{x}) = f\left(\mathbf{w}^T \boldsymbol{\phi}(\mathbf{x})\right) \tag{2.4}$$

dove:

- $\mathbf{w} = (w_1, w_2, \ldots, w_n) \in \mathbb{R}^n$ è il vettore dei **pesi**
- $f(\cdot)$ è la **funzione di attivazione non lineare**

Il vettore di feature $\boldsymbol{\phi}(\mathbf{x})$ include una componente di bias $\phi_0(\mathbf{x}) = 1$, consentendo di incorporare il bias (termine costante) direttamente nel vettore dei pesi, semplificando la notazione matematica.

Nel perceptron originale di Rosenblatt, la funzione di attivazione è data da una funzione a gradino (step function):

$$f(a) = \begin{cases} +1 & \text{se } a \geq 0 \\ -1 & \text{se } a < 0 \end{cases} \tag{2.5}$$

Questa scelta di funzione di attivazione rende il perceptron un classificatore binario: l'output è $+1$ per la classe $\mathcal{C}_1$ e -1 per la classe $\mathcal{C}_2$.

Interpretazione geometrica. Nel caso della funzione di attivazione a gradino, il perceptron implementa un classificatore lineare che divide lo spazio degli input in due regioni separate da un iperpiano decisionale definito da $\mathbf{w}^T \boldsymbol{\phi}(\mathbf{x}) = 0$ [9]. Punti che giacciono su un lato dell'iperpiano vengono assegnati alla classe $\mathcal{C}_1$ ($y = +1$), mentre punti sull'altro lato vengono assegnati alla classe $\mathcal{C}_2$ ($y = -1$).

Questa limitazione a problemi linearmente separabili venne identificata da Minsky e Papert [38] come una debolezza fondamentale del perceptron singolo. Il loro lavoro dimostrò che esistono problemi semplici, come la funzione XOR, che non possono essere risolti da un singolo percettrone. Questa limitazione può essere superata solo attraverso architetture multi-layer che combinano molteplici perceptron in strutture stratificate.

Dal Percettrone al Multi-Layer Perceptron

Per superare le limitazioni del perceptron singolo, è necessario organizzare molteplici perceptron in **layer** (strati) successivi, dando origine al **Multi-Layer Perceptron** (MLP) o **rete neurale feedforward** [19]. Questa architettura introduce due innovazioni fondamentali rispetto al perceptron: (1) la presenza di uno o più hidden layer tra input e output, e (2) l'utilizzo di funzioni di attivazione differenziabili al posto della step function, necessarie per consentire l'apprendimento tramite backpropagation [39].

Architettura. Un MLP è composto da tre tipi di layer [9], [19]:

- **Input layer**: riceve i dati grezzi rappresentati come vettore $\mathbf{x} \in \mathbb{R}^d$, dove d è la dimensionalità dell'input. Questo layer non effettua computazioni ma distribuisce semplicemente l'input ai layer successivi, eventualmente dopo una trasformazione $\boldsymbol{\phi}(\mathbf{x})$.
- **Hidden layers**: uno o più layer intermedi che trasformano progressivamente l'input. Ogni hidden layer apprende una rappresentazione dei dati a un diverso livello di astrazione. Come discusso nella Sezione 2.2.1, i layer iniziali tendono a estrarre feature di basso livello, mentre layer più profondi costruiscono rappresentazioni sempre più astratte [31].
- **Output layer**: produce la predizione finale. Per problemi di classificazione con C classi, tipicamente contiene C neuroni con funzione di attivazione softmax che produce una distribuzione di probabilità sulle classi. Per problemi di regressione, può contenere uno o più neuroni con attivazione lineare.

Formulazione Matematica di una Rete Neurale a Due Layer. Seguendo la notazione di Bishop [9], le reti neurali possono essere interpretate come un'estensione parametrica dei modelli lineari per regressione e classificazione. Nei modelli lineari tradizionali, l'output è ottenuto come combinazione lineare di funzioni base fisse $\phi_j(\mathbf{x})$:

$$y(\mathbf{x}, \mathbf{w}) = f\left(\sum_{j=1}^{M} w_j \phi_j(\mathbf{x})\right) \tag{2.6}$$

dove $f(\cdot)$ è una funzione di attivazione non lineare (identità per la regressione, sigmoide per la classificazione). La rivoluzione introdotta dalle reti neurali consiste nel rendere anche le funzioni base $\phi_j(\mathbf{x})$ stesse parametriche e adattabili durante l'addestramento, costruendo così rappresentazioni gerarchiche dei dati.

Consideriamo una rete neurale feedforward con un singolo layer nascosto. Il primo layer computa M combinazioni lineari delle variabili di input $x_1, \ldots, x_D$:

$$a_j = \sum_{i=1}^{D} w_{ji}^{(1)} x_i + w_{j0}^{(1)}, \quad j = 1, \ldots, M \tag{2.7}$$

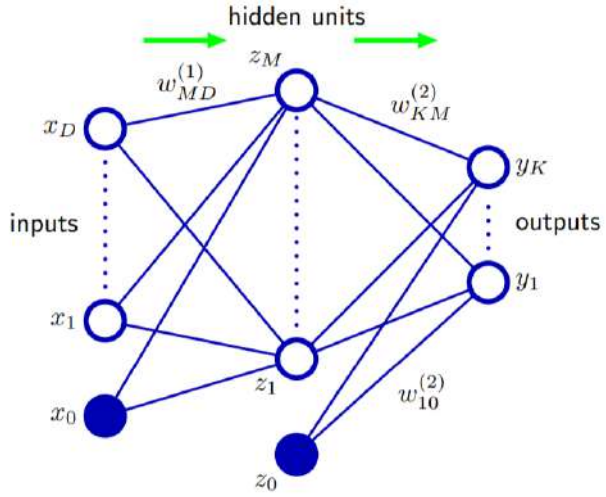

Figure 2.6: Architettura di una rete neurale feedforward a due layer con un singolo strato nascosto. I nodi di input $x_0, x_1, \ldots, x_D$ sono collegati alle unità nascoste $z_1, \ldots, z_M$ attraverso pesi $w^{(1)}_{ji}$. L'unità x_0 rappresenta il bias per lo strato nascosto. Le unità nascoste sono poi collegate alle unità di output $y_1, \ldots, y_K$ attraverso pesi $w^{(2)}_{kj}$, con z_0 come bias per lo strato di output. Adattato da [9, Fig. 5.1].

dove $w^{(1)}_{ji}$ sono i pesi del primo layer, $w^{(1)}_{j0}$ sono i bias, e a_j sono le attivazioni lineari. Queste vengono poi trasformate attraverso una funzione di attivazione differenziabile e non lineare $h(\cdot)$:

$$z_j = h(a_j) \tag{2.8}$$

Le quantità z_j sono dette *unità nascoste* e corrispondono alle funzioni base adattive della rete. Infine, le unità nascoste sono combinate linearmente per produrre le attivazioni di output:

$$a_k = \sum_{j=1}^{M} w^{(2)}_{kj} z_j + w^{(2)}_{k0}, \quad k = 1, \ldots, K \tag{2.9}$$

dove K è il numero di output. L'output finale della rete si ottiene applicando un'ulteriore funzione di attivazione appropriata al compito (es. softmax per classificazione multiclasse):

$$y_k = g(a_k) \tag{2.10}$$

Questa composizione di trasformazioni affini e non lineari consente alla rete di apprendere funzioni arbitrariamente complesse, superando la limitazione del perceptron singolo ai soli problemi linearmente separabili.

Necessità di funzioni differenziabili. A differenza del perceptron originale che utilizza una step function, i MLP moderni richiedono funzioni di attivazione differenziabili. Questo perché l'apprendimento avviene tramite l'algoritmo di backpropagation, che calcola i gradienti della funzione di errore rispetto ai parametri della rete propagando l'errore all'indietro attraverso i layer [39]. La step function, avendo gradiente zero quasi ovunque (eccetto nel punto di discontinuità), non è adatta a questo scopo. Nella sezione successiva esamineremo le principali funzioni di attivazione differenziabili utilizzate nelle reti neurali moderne.

Funzioni di Attivazione

Come anticipato, la scelta della funzione di attivazione $h(\cdot)$ negli strati nascosti e $g(\cdot)$ nello strato di output è cruciale per il successo dell'addestramento di una rete neurale. Essa deve soddisfare due requisiti fondamentali [19]:

- introdurre **non linearità** per consentire l'approssimazione di funzioni complesse (senza non linearità, anche una rete con molteplici layer si ridurrebbe a una semplice trasformazione lineare)
- essere **differenziabile** (o almeno avere gradiente ben definito quasi ovunque) per permettere l'ottimizzazione tramite discesa del gradiente e backpropagation

La ricerca di funzioni che bilanciano espressività, stabilità nell'apprendimento e efficienza computazionale ha portato all'adozione di diverse famiglie di funzioni.

Le funzioni di attivazione più utilizzate includono:

Sigmoid. La funzione sigmoide (o logistica) è stata storicamente una delle prime funzioni di attivazione utilizzate nelle reti neurali [39]:

$$\sigma(a) = \frac{1}{1 + e^{-a}} \tag{2.11}$$

Mappa l'input nell'intervallo $(0, 1)$, rendendola naturalmente adatta per interpretazioni probabilistiche. La sua derivata ha la forma $\sigma'(a) = \sigma(a)(1-\sigma(a))$, utile per il calcolo dei gradienti durante la backpropagation.

Tuttavia, presenta il problema del *vanishing gradient*: per valori di input molto grandi ($a \gg 0$) o molto piccoli ($a \ll 0$), la derivata tende a zero e il gradiente si annulla, rallentando significativamente l'apprendimento nei layer più profondi [40], [41].

Tangente Iperbolica (tanh). Simile alla sigmoid ma con output nell'intervallo $(-1, 1)$:

$$\tanh(a) = \frac{e^{a} - e^{-a}}{e^{a} + e^{-a}} = 2\sigma(2a) - 1 \tag{2.12}$$

Rispetto alla sigmoid, la tanh produce output centrati sullo zero, caratteristica che facilita l'ottimizzazione poiché i gradienti tendono a non saturare prematuramente [42]. Tuttavia, soffre anch'essa del problema del vanishing gradient per valori estremi.

ReLU (Rectified Linear Unit). La funzione ReLU è diventata lo standard de facto nelle reti neurali profonde [43], [44]:

$$\text{ReLU}(a) = \max(0, a) = \begin{cases} a & \text{se } a > 0 \\ 0 & \text{se } a \leq 0 \end{cases} \tag{2.13}$$

Presenta diversi vantaggi:

- **Efficienza computazionale**: richiede solo un confronto e una selezione, senza operazioni esponenziali costose
- **Mitigazione del vanishing gradient**: per input positivi, il gradiente è costante e uguale a 1, facilitando la propagazione del segnale di errore [44]
- **Sparsità**: neuroni con input negativi vengono "spenti" (attivazione zero), inducendo rappresentazioni sparse. Questa sparsità può migliorare l'efficienza computazionale e potenzialmente la capacità di generalizzazione della rete [44]

Varianti di ReLU. Tuttavia, la ReLU presenta il problema dei *neuroni morti* (dying ReLU): se un neurone riceve costantemente input negativi, il gradiente è sempre zero e il neurone non si aggiorna più, diventando permanentemente inattivo [45]. Per mitigare questo problema, sono state proposte diverse varianti della funzione.

La **Leaky ReLU** [46] introduce una pendenza minima α (tipicamente 0.01) per gli input negativi:

$$\text{LeakyReLU}(x) = \max(\alpha x, x), \quad \alpha \in (0, 1)$$

Questa modifica garantisce che il gradiente sia sempre non nullo ($\nabla_x = \alpha$ per $x < 0$), prevenendo la morte permanente dei neuroni.

Una generalizzazione più flessibile è la **Parametric ReLU (PReLU)** [45], in cui il coefficiente α non è fissato a priori ma viene appreso durante l'addestramento come parametro aggiuntivo della rete:

$$\text{PReLU}(x) = \max(\alpha x, x), \quad \alpha \text{ imparabile}$$

PReLU permette a ciascun layer (o addirittura a ciascun neurone) di adattare autonomamente la risposta agli input negativi, bilanciando l'azione di attivazione e regolarizzazione in base alle specificità dei dati. In pratica, la rete può decidere se comportarsi più come una ReLU standard ($\alpha \approx 0$) o come una Leaky ReLU con pendenza significativa ($\alpha > 0$), ottimizzando questo trade-off per il task specifico.

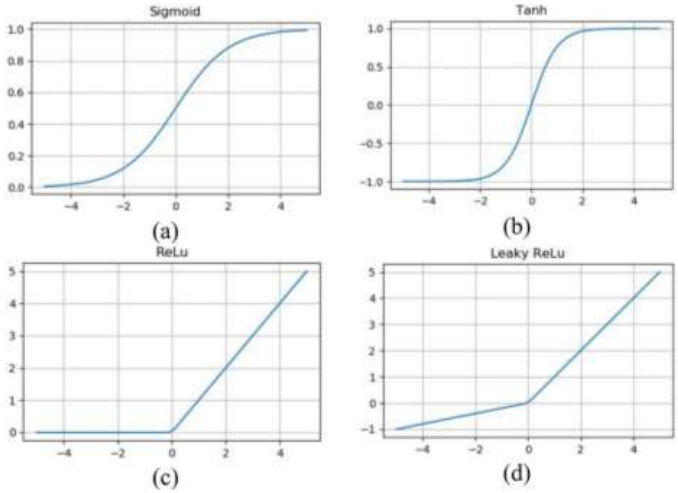

Figure 2.7: Confronto delle principali funzioni di attivazione utilizzate nelle reti neurali: (a) Sigmoid, (b) Tanh, (c) ReLU, (d) Leaky ReLU. Fonte: [47].

La scelta della funzione di attivazione può avere un impatto significativo sulle prestazioni della rete. In generale, la ReLU e le sue varianti sono preferite per i layer nascosti nelle architetture profonde moderne, mentre la scelta per il layer di output dipende dal task specifico: softmax per classificazione multi-classe, sigmoid per classificazione binaria, e identità (lineare) per regressione.

Architetture di Reti Neurali

Le reti neurali possono essere classificate in base alla direzione del flusso dell'informazione attraverso la rete. Le due architetture fondamentali sono le reti **feedforward** e le reti **ricorrenti** (recurrent) [19].

Reti Feedforward. Nelle reti feedforward, l'informazione fluisce in una sola direzione, dall'input verso l'output, senza cicli o connessioni retroattive. Queste reti sono rappresentate da grafi aciclici diretti (Directed Acyclic Graphs, DAG), dove ogni layer riceve input solo dai layer precedenti e non esiste alcun feedback [9].

Il Multi-Layer Perceptron (MLP) discusso in 2.2.2 è l'esempio fondamentale di architettura feedforward, in cui ogni layer è completamente connesso (*fully-connected* o *dense*) al layer successivo. In un MLP, ogni neurone in un layer riceve input da tutti i neuroni del layer precedente, risultando in un numero di parametri pari a $n_{l-1} \times n_l$ per la connessione tra due layer consecutivi con n_{l-1} e n_l neuroni rispettivamente.

Altre importanti architetture feedforward includono le **reti convoluzionali** (Convolutional Neural Networks, CNN), che introducono due principi fondamentali:

- **Connettività locale**: ogni neurone è connesso solo a una regione locale dell'input, riducendo drasticamente il numero di parametri.
- **Condivisione dei pesi**: gli stessi pesi (filtri) sono applicati a diverse posizioni spaziali, sfruttando l'invarianza traslazionale dei dati visivi.

Queste caratteristiche rendono le CNN particolarmente efficaci per dati con struttura spaziale, come immagini, dove pattern locali (bordi, texture) e gerarchie spaziali sono fondamentali. Le CNN saranno discusse in dettaglio nella sezione successiva.

Le reti feedforward sono particolarmente adatte per problemi in cui l'output dipende solo dall'input corrente, senza necessità di mantenere informazioni su input precedenti. Esempi tipici includono:

- Classificazione di immagini statiche
- Regressione su dati tabulari
- Riconoscimento di pattern in cui non è presente una struttura temporale o sequenziale

Reti Ricorrenti. Le reti ricorrenti (Recurrent Neural Networks, RNN), al contrario, contengono connessioni cicliche che consentono all'informazione di persistere nel tempo. Le attivazioni possono propagarsi sia in avanti che all'indietro, creando una sorta di "memoria" interna che permette alla rete di elaborare sequenze di input [19]. Formalmente, lo stato nascosto $\mathbf{h}^{(t)}$ al tempo t è funzione sia dell'input corrente $\mathbf{x}^{(t)}$ sia dello stato precedente $\mathbf{h}^{(t-1)}$:

$$\mathbf{h}^{(t)} = f(\mathbf{h}^{(t-1)}, \mathbf{x}^{(t)}; \theta) \tag{2.14}$$

dove f è una trasformazione non lineare e θ rappresenta l'insieme dei parametri della rete. Nella sua forma più comune, detta RNN "vanilla" o "Elman network" [48], questa trasformazione si realizza come combinazione lineare seguita da una non-linearità:

$$\mathbf{h}^{(t)} = \tanh(\mathbf{W}_{hh}\mathbf{h}^{(t-1)} + \mathbf{W}_{xh}\mathbf{x}^{(t)} + \mathbf{b}_h) \tag{2.15}$$

dove $\mathbf{W}_{hh}$ e $\mathbf{W}_{xh}$ sono matrici di pesi e $\mathbf{b}_h$ è un vettore bias. L'output al tempo t si ottiene trasformando lo stato nascosto:

$$\mathbf{y}^{(t)} = g(\mathbf{W}_{hy}\mathbf{h}^{(t)} + \mathbf{b}_y) \tag{2.16}$$

dove $g(\cdot)$ è la funzione di attivazione dell'output.

Questa architettura permette alla rete di catturare dipendenze temporali, poiché lo stato $\mathbf{h}^{(t)}$ funge da memoria compressa della sequenza osservata fino al tempo t. Tuttavia, le RNN standard soffrono del *vanishing gradient problem* [41], che rende difficile apprendere dipendenze a lungo termine: durante la backpropagation attraverso il tempo (BPTT), i gradienti tendono a svanire (o esplodere) quando retro-propagati attraverso molti step temporali.

Per mitigare questo problema, sono state sviluppate nuove architetture con meccanismi di *gating*:

- **LSTM (Long Short-Term Memory)** [49]: Introduce tre gate (input, forget, output) e una cell state che permettono un controllo preciso del flusso di informazione, facilitando la memorizzazione a lungo termine.
- **GRU (Gated Recurrent Unit)** [50]: Variante semplificata con due gate (reset e update), che offre prestazioni simili alle LSTM con meno parametri.

Le RNN e le loro varianti trovano applicazione in numerosi domini che coinvolgono dati sequenziali, tra cui: elaborazione del linguaggio naturale (NLP), riconoscimento vocale, analisi di serie temporali, e generazione di contenuti sequenziali (testo, musica, video).

2.2.3 Training delle Reti Neurali

Una volta definita l'architettura di una rete neurale, è necessario determinare i valori ottimali dei parametri (pesi e bias) che minimizzano l'errore sulle predizioni. Questo processo, detto **training** o **addestramento**, si basa sull'ottimizzazione iterativa dei parametri attraverso algoritmi basati sul gradiente.

Backpropagation

L'algoritmo di **backpropagation** (retropropagazione dell'errore) [39], rappresenta il metodo fondamentale per calcolare i gradienti della funzione di errore rispetto a tutti i parametri di una rete neurale. La sua formulazione ha reso possibile l'addestramento di reti profonde.

Formulazione matematica. Consideriamo una rete neurale feedforward in cui le unità sono organizzate in layer. Seguendo la notazione di Bishop [9], per ogni unità j nella rete definiamo:

- a_j: la **pre-attivazione**, data dalla somma pesata degli input all'unità j
- z_j: l'**attivazione**, ottenuta applicando la funzione di attivazione: $z_j = h(a_j)$
- w_{ji}: il peso della connessione dall'unità i all'unità j

dove $h(\cdot)$ è una funzione di attivazione differenziabile (ad esempio sigmoid, tanh, o ReLU).

L'obiettivo della backpropagation è calcolare efficientemente i gradienti $\frac{\partial E}{\partial w_{ji}}$ della funzione di errore E rispetto a tutti i pesi della rete.

Errori e gradienti. Definiamo l'**errore** δ_j per l'unità j come la derivata parziale della funzione di errore rispetto alla sua pre-attivazione [9]:

$$\delta_j \equiv \frac{\partial E}{\partial a_j} \tag{2.17}$$

Applicando la regola della catena (*chain rule*), il gradiente rispetto al peso w_{ji} può essere espresso come:

$$\frac{\partial E}{\partial w_{ji}} = \frac{\partial E}{\partial a_j} \cdot \frac{\partial a_j}{\partial w_{ji}} \tag{2.18}$$

Poiché $a_j = \sum_i w_{ji} z_i$, si ha $\frac{\partial a_j}{\partial w_{ji}} = z_i$, da cui:

$$\frac{\partial E}{\partial w_{ji}} = \delta_j \, z_i \tag{2.19}$$

Questo risultato fondamentale mostra che il gradiente rispetto a un peso è dato dal prodotto dell'errore δ_j dell'unità di destinazione e dell'attivazione z_i dell'unità di origine. Analogamente, per i bias si ottiene:

$$\frac{\partial E}{\partial b_j} = \delta_j \tag{2.20}$$

Calcolo ricorsivo degli errori. La potenza della backpropagation risiede nel calcolo efficiente degli errori δ_j tramite una procedura ricorsiva che propaga gli errori all'indietro attraverso la rete.

Per le unità di **output**, l'errore dipende dalla funzione di loss utilizzata. Per l'errore quadratico (sum-of-squares):

$$E = \frac{1}{2} \sum_k (y_k - t_k)^2 \tag{2.21}$$

dove y_k è l'output dell'unità k e t_k il target, si ottiene direttamente [9]:

$$\delta_k = \frac{\partial E}{\partial a_k} = \frac{\partial E}{\partial y_k} \cdot \frac{\partial y_k}{\partial a_k} = (y_k - t_k)\, h'(a_k) \tag{2.22}$$

Per le unità negli **hidden layer**, si applica nuovamente la chain rule. L'errore di un'unità j in un hidden layer dipende dagli errori di tutte le unità k a cui j è connessa nel layer successivo:

$$\delta_j = \frac{\partial E}{\partial a_j} = \sum_k \frac{\partial E}{\partial a_k} \cdot \frac{\partial a_k}{\partial a_j} \tag{2.23}$$

Poiché $a_k = \sum_j w_{kj} z_j = \sum_j w_{kj} h(a_j)$, si ha $\frac{\partial a_k}{\partial a_j} = w_{kj} h'(a_j)$, da cui [9]:

$$\delta_j = h'(a_j) \sum_k w_{kj} \, \delta_k \tag{2.24}$$

Questa equazione mostra che l'errore in un hidden layer si ottiene sommando gli errori del layer successivo pesati dai corrispondenti pesi di connessione, e moltiplicando per la derivata della funzione di attivazione.

Algoritmo di backpropagation. L'algoritmo completo procede in due fasi distinte per ciascun esempio (o mini-batch) presentato alla rete [9], [39]:

1. **Forward pass**: si applica un pattern di input alla rete e si propagano i segnali in avanti attraverso i layer, calcolando le pre-attivazioni a_j e le attivazioni $z_j = h(a_j)$ di tutte le unità fino agli output. Si valuta quindi la funzione di errore E confrontando gli output della rete $\mathbf{y}$ con i target desiderati $\mathbf{t}$.
2. **Backward pass**: si calcolano gli errori δ propagando all'indietro attraverso la rete:
   - Per le unità di output, si calcola δ_k usando l'Equazione 2.22
   - Per ciascun hidden layer, procedendo dall'ultimo verso il primo, si calcolano i δ_j usando l'Equazione 2.24
   - Una volta noti tutti i δ_j, si calcolano i gradienti dei pesi tramite l'Equazione 2.19

Questi gradienti, accumulati su un mini-batch, vengono infine utilizzati da un algoritmo di ottimizzazione basato sulla discesa del gradiente per aggiornare tutti i parametri della rete. L'intero processo (forward pass, backward pass, aggiornamento dei pesi) viene iterato per molte epoche su tutto il dataset di addestramento, guidando la rete verso un minimo della funzione di errore.

Efficienza computazionale. La backpropagation è molto efficiente: richiede circa il doppio delle operazioni della forward propagation. Per ogni connessione nella rete, l'algoritmo esegue una moltiplicazione nel forward pass e una nel backward pass, più un'addizione. Questa efficienza è fondamentale per rendere praticabile l'addestramento di reti con milioni di parametri.

Importanza della differenziabilità. L'algoritmo richiede che le funzioni di attivazione siano differenziabili, poiché la backpropagation si basa sul calcolo delle derivate $h'(a_j)$. Questo spiega perché la funzione a gradino del perceptron originale, avendo derivata zero quasi ovunque, impedisce l'uso della backpropagation. Le funzioni moderne come sigmoid, tanh e ReLU, essendo differenziabili (o sub-differenziabili nel caso della ReLU nel punto zero), permettono il calcolo efficiente dei gradienti.

Estensione a reti profonde. Sebbene abbiamo descritto l'algoritmo senza esplicitare i layer con notazione indicizzata, l'estensione a reti con molti layer è immediata: si applica ricorsivamente l'Equazione 2.24 procedendo all'indietro dal layer di output fino al primo hidden layer. Questa applicazione ricorsiva

della chain rule attraverso i layer è ciò che rende possibile l'apprendimento in reti profonde [19].

Funzioni di Errore

La scelta della **funzione di errore** (o *loss function*) è importante per l'addestramento di una rete neurale perché definisce l'obiettivo che l'ottimizzazione cerca di minimizzare. La funzione di errore deve essere differenziabile per permettere il calcolo dei gradienti tramite backpropagation e deve riflettere il tipo di problema affrontato [9].

Errore Quadratico Medio. Per problemi di **regressione**, dove l'obiettivo è predire valori continui, la funzione di errore più comune è l'**errore quadratico** (sum-of-squares error) o la sua media, il *Mean Squared Error* (MSE) [12]. Data una rete con K output e N esempi di training, l'errore totale è definito come [9]:

$$E = \frac{1}{2N} \sum_{n=1}^{N} \sum_{k=1}^{K} (y_{nk} - t_{nk})^2 \tag{2.25}$$

dove y_{nk} è l'output della rete per l'esempio n e l'unità di output k, e t_{nk} è il corrispondente valore target. Il fattore $\frac{1}{2}$ è una convenzione che semplifica la derivata.

Sotto l'assunzione che i target siano corrotti da rumore gaussiano, minimizzare l'MSE equivale a massimizzare la log-likelihood dei dati, fornendo una giustificazione probabilistica per questa scelta [9]. Il gradiente dell'MSE rispetto all'output è:

$$\frac{\partial E}{\partial y_{nk}} = y_{nk} - t_{nk} \tag{2.26}$$

che si combina con la derivata della funzione di attivazione durante la backpropagation.

Cross-Entropy per Classificazione. Per problemi di classificazione, la funzione di errore standard è la **cross-entropy**, che misura la divergenza tra la distribuzione di probabilità predetta dalla rete e la distribuzione reale dei target [9], [19].

Classificazione Binaria. Con target $t_n \in \{0, 1\}$ e output sigmoid $y_n \in (0, 1)$, la binary cross-entropy è:

$$E = -\sum_{n=1}^{N} [t_n \log y_n + (1 - t_n) \log(1 - y_n)]$$

Classificazione Multi-Classe. Con target one-hot $\mathbf{t}_n \in \{0,1\}^C$ e output softmax $\mathbf{y}_n \in (0,1)^C$ (dove $\sum_k y_{nk} = 1$), la categorical cross-entropy è:

$$E = -\sum_{n=1}^{N}\sum_{k=1}^{C} t_{nk} \log y_{nk}$$

In entrambi i casi, il gradiente rispetto alle pre-attivazioni dello strato di output si semplifica in $\frac{\partial E}{\partial a_k} = y_k - t_k$, facilitando l'implementazione della backpropagation [19].

Scelta della Funzione di Errore. La scelta della funzione di errore deve riflettere la natura del problema:

- **Regressione**: MSE (o varianti come Mean Absolute Error per robustezza agli outlier)
- **Classificazione binaria**: Binary cross-entropy con sigmoid
- **Classificazione multi-classe**: Categorical cross-entropy con softmax
- **Multi-label classification**: Binary cross-entropy per ciascuna label (le classi non sono mutualmente esclusive)

La cross-entropy è generalmente preferita all'MSE per problemi di classificazione perché:

- Fornisce gradienti più informativi quando l'output è lontano dal target
- Ha una giustificazione probabilistica diretta (massimizzazione della likelihood)
- Evita problemi di saturazione che possono verificarsi con MSE e funzioni di attivazione sigmoidali

Table 2.1: Funzioni di errore per diversi tipi di problema.

Tipo	**Formula**	**Attivazione output**
Regressione	$\frac{1}{2}\sum(y-t)^2$	Identità
Classificazione binaria	$-\sum[t\log y + (1-t)\log(1-y)]$	Sigmoid
Classificazione multi-classe	$-\sum\sum_k t_k \log y_k$	Softmax

Ottimizzazione basata sul Gradiente

Una volta calcolati i gradienti tramite backpropagation, è necessario utilizzarli per aggiornare i parametri della rete. Gli algoritmi di ottimizzazione determinano come modificare i pesi in funzione dei gradienti per minimizzare la funzione di errore [19].

Mini-batch Stochastic Gradient Descent (SGD). Il gradient descent classico [9] aggiorna i parametri nella direzione opposta al gradiente calcolato sull'intero dataset. Per dataset di grandi dimensioni, questa operazione è computazionalmente proibitiva. Lo **Stochastic Gradient Descent (SGD)** [51] utilizza invece il gradiente calcolato su un singolo esempio o, più comunemente, su un **mini-batch** di dimensione ridotta (tipicamente 32-256 esempi). Questo approccio permette aggiornamenti più frequenti e sfrutta efficientemente l'elaborazione parallela delle GPU.

SGD con Momentum. Per mitigare le oscillazioni di SGD in funzioni di errore con geometrie complesse (come valli strette o plateau), si introduce il **momentum** [52], [53]. Questo concetto, ispirato alla fisica, mantiene una "velocità" di aggiornamento che accumula il contributo dei gradienti passati, permettendo di accelerare la convergenza lungo direzioni con gradiente costante e riducendo le oscillazioni nelle direzioni ortogonali.

Adam (Adaptive Moment Estimation). L'ottimizzatore **Adam** [54] combina i vantaggi del momentum con l'adattività del learning rate per ciascun parametro. Mantiene due statistiche in esecuzione: una media mobile esponenziale dei gradienti (primo momento) e dei loro quadrati (secondo momento). Questa stima permette di adattare dinamicamente la dimensione del passo di aggiornamento, risultando robusto alla scelta del learning rate iniziale e convergendo rapidamente in molti scenari pratici. Adam è diventato uno degli ottimizzatori più utilizzati nel deep learning.

Learning Rate Scheduling. Indipendentemente dall'ottimizzatore scelto, la performance è spesso migliorata riducendo strategicamente il learning rate durante il training. Strategie comuni includono **step decay** (riduzione periodica), l'**exponential decay** e il **cosine annealing** [19]. Tecniche come il **warmup** (aumento graduale del learning rate all'inizio) possono anche stabilizzare le fasi iniziali dell'addestramento.

Regolarizzazione

Le tecniche di regolarizzazione sono fondamentali per prevenire l'overfitting e migliorare la capacità di generalizzazione delle reti a dati non visti. L'overfitting si verifica quando il modello si adatta eccessivamente ai dati di training, catturando anche il rumore anziché solo i pattern rilevanti, risultando in prestazioni scarse sul test set.

L2 Regularization (Weight Decay). La regolarizzazione L2, o weight decay, aggiunge alla funzione di errore un termine di penalizzazione proporzionale al quadrato dei pesi [9]:

$$\tilde{E}(\mathbf{w}) = E(\mathbf{w}) + \frac{\lambda}{2}\|\mathbf{w}\|^2$$

dove $\lambda > 0$ è il coefficiente di regolarizzazione che controlla l'intensità della penalizzazione. Questa penalizzazione favorisce pesi di piccola magnitudine, promuovendo modelli più semplici e meno inclini a sovradattarsi.

Dropout. Il dropout [55] è una tecnica stocastica che, durante il training, "disattiva" casualmente una frazione dei neuroni (tipicamente il 50%). Questo obbliga la rete a sviluppare rappresentazioni ridondanti e robuste, migliorando la generalizzazione. Concettualmente, il dropout può essere visto come il training di un ensemble di reti diverse che vengono combinate a test time.

Batch Normalization. La batch normalization [56] normalizza le attivazioni di ciascun layer per avere media zero e varianza unitaria su ogni mini-batch. Questo stabilizza la distribuzione degli input ai layer successivi, permettendo learning rate più alti, accelerando la convergenza e fornendo anche un leggero effetto regolarizzante.

Early Stopping. L'early stopping [57] interrompe il training quando la performance su un validation set smette di migliorare, prevenendo così l'overfitting. È una forma di regolarizzazione semplice ma efficace, particolarmente utile quando la capacità del modello è elevata rispetto alla complessità del problema.

Problemi Comuni e Soluzioni

Vanishing e Exploding Gradients. Nelle reti profonde, i gradienti possono diventare estremamente piccoli (vanishing) o grandi (exploding) durante la back-propagation [40], [41]. Questo fenomeno, legato alla moltiplicazione ripetuta di derivate attraverso molti layer, può impedire l'apprendimento efficace.

Soluzioni principali includono:

- **Funzioni di attivazione** come ReLU, che hanno derivata non nulla per input positivi.
- **Inizializzazioni appropriate** dei pesi (es., Xavier [42] o He [45]).
- **Batch Normalization**, che stabilizza la distribuzione delle attivazioni.
- **Skip connections** come nelle ResNet [58], che permettono ai gradienti di fluire più facilmente.

Scelta degli Iperparametri. Il training efficace richiede la sintonizzazione di numerosi iperparametri (learning rate, dimensione del batch, coefficienti di regolarizzazione, ecc.). Pratiche comuni includono la **grid search** o **random search** [59] sullo spazio degli iperparametri, validate tramite **cross-validation**. Il monitoraggio delle **learning curves** (loss su training e validation set) è essenziale per diagnosticare overfitting o underfitting e guidare le regolazioni.

2.3 Convolutional Neural Networks

Le **Convolutional Neural Networks** (CNN, reti neurali convoluzionali) rappresentano una classe specializzata di reti neurali feedforward progettata per processare dati con struttura a griglia, come le immagini. Introdotte da LeCun et al. [60] e rese popolari dal successo su ImageNet di Krizhevsky et al. [61], le CNN hanno rivoluzionato il campo della visione artificiale sfruttando le proprietà caratteristiche dei dati visivi: connettività locale, invarianza traslazionale e struttura gerarchica.

Motivazione: Limitazioni degli MLP per le Immagini

Applicare un Multi-Layer Perceptron fully-connected direttamente alle immagini presenta problemi fondamentali che rendono questo approccio impraticabile:

Numero di parametri e scalabilità. Un MLP fully-connected applicato direttamente alle immagini soffre di problemi di scalabilità dovuti al numero eccessivo di parametri. Come osservato da [19], ogni unità in un hidden layer deve avere un peso separato per ogni pixel dell'immagine di input.

Consideriamo un esempio concreto: un'immagine RGB di dimensione 224×224 pixels (la dimensione standard utilizzata per il training su ImageNet [62]) contiene $224 \times 224 \times 3 = 150,528$ valori di input. Un primo hidden layer con soli 1000 neuroni richiederebbe $150,528 \times 1000 \approx 150$ milioni di parametri. Le conseguenze sono molteplici:

- **Costo computazionale**: il training e l'inferenza diventano eccessivamente costosi in termini di memoria e tempo di calcolo
- **Overfitting**: data l'enorme quantità di parametri da stimare, il modello tende facilmente a memorizzare il training set
- **Necessità di dataset** estremamente grandi per l'addestramento efficace

Questo problema si aggrava ulteriormente per immagini di risoluzione superiore o per architetture con molteplici hidden layer.

Mancanza di invarianza spaziale. In un MLP, un pattern visivo (ad esempio un bordo verticale o una texture) che appare in posizioni diverse dell'immagine viene trattato come feature indipendenti [19]. La rete deve imparare separatamente a riconoscere lo stesso pattern in ogni possibile posizione e questo rappresenta uno spreco di capacità rappresentativa. Questa mancanza di *parameter sharing* (condivisione dei parametri) impedisce alla rete di generalizzare pattern appresi da una regione dell'immagine ad altre regioni.

Perdita della struttura spaziale. Trasformare un'immagine bidimensionale in un vettore unidimensionale per l'input a un MLP distrugge l'organizzazione spaziale 2D originale. Sebbene l'ordine dei pixel nel vettore possa conservare una qualche struttura (ad esempio, per righe successive), l'MLP non ha alcun meccanismo per sfruttare questa organizzazione: ogni neurone del primo strato è connesso a tutti i pixel del vettore, senza distinguere tra pixel vicini o lontani nello spazio originale.

Tuttavia, nelle immagini naturali, la vicinanza spaziale è fortemente informativa: pixel adiacenti sono tipicamente correlati e la composizione locale di pixel forma pattern significativi come bordi, angoli e texture. Questa struttura viene ignorata dagli MLP ma è invece fondamentale per le CNN, che sono appositamente progettate per rilevare pattern locali attraverso i filtri convoluzionali.

Le CNN risolvono sistematicamente questi problemi attraverso tre principi architetturali chiave [30]: **connettività locale**, **condivisione dei pesi**, e **pooling**.

Principi Fondamentali delle CNN

Connettività locale. Invece di connettere ogni neurone a tutti i pixel dell'immagine, le CNN sfruttano la **connettività locale**: ciascuna unità in un layer convoluzionale riceve input solo da una piccola regione del layer precedente, detta *receptive field* locale. Ad esempio, un filtro 3×3 connette ogni neurone a sole 9 posizioni spaziali.

Questo riflette la natura locale delle correlazioni nelle immagini naturali: per comprendere cosa rappresenta un pixel, è tipicamente sufficiente esaminare i suoi vicini. La connettività locale riduce drasticamente il numero di parametri, rendendo le CNN scalabili a immagini ad alta risoluzione.

Condivisione dei pesi (Weight Sharing). Un aspetto rivoluzionario delle CNN è che lo stesso filtro (gli stessi pesi) viene applicato a tutte le posizioni spaziali dell'input. Questo meccanismo è detto **weight sharing**:

- Un filtro di dimensione 3×3 applicato a un'immagine 224×224 ha solo 9 parametri, indipendentemente dalla dimensione dell'input
- **Equivarianza traslazionale:** la rete può riconoscere lo stesso pattern (ad esempio un bordo verticale) indipendentemente dalla sua posizione nell'immagine
- Il numero totale di parametri è ridotto: applicare 64 filtri 3×3 richiede solo $64 \times 3 \times 3 = 576$ parametri, contro i milioni di un layer fully-connected equivalente

Concettualmente, ciascun filtro impara a rilevare una specifica caratteristica visiva (un bordo, una texture, un pattern di colore), e la condivisione dei pesi garantisce che questa caratteristica possa essere rilevata ovunque appaia nell'immagine.

Operazione di convoluzione. L'operazione di convoluzione consiste nell'applicare un filtro (*kernel*) di piccole dimensioni che scorre sull'immagine di input. In ogni posizione, il filtro calcola il prodotto elemento per elemento con la regione sottostante, seguito da una somma. Il risultato è una *feature map* che indica quanto il pattern cercato dal filtro è presente in ciascuna posizione.

Un layer convoluzionale applica tipicamente molteplici filtri in parallelo, producendo diverse feature map [30]. Se l'input ha C_{in} canali (ad esempio 3 per un'immagine RGB) e si applicano C_{out} filtri, ciascuno di dimensione $h \times w \times C_{in}$, si ottengono C_{out} feature map di output.

Pooling. Il **pooling** (o *subsampling*) è un'operazione che riduce progressivamente le dimensioni spaziali delle feature map, diminuendo il numero di parametri nei layer successivi e la complessità computazionale. L'operazione più comune è il **max pooling**, che divide l'input in regioni non sovrapposte (tipicamente 2×2) e seleziona il valore massimo in ciascuna regione. Un max pooling 2×2 con stride 2 riduce le dimensioni dell'output della metà.

Il pooling offre diversi vantaggi: introduce una forma di invarianza a piccole traslazioni, aumenta il receptive field effettivo dei layer successivi, e aiuta a prevenire overfitting riducendo la complessità del modello [19].

Architettura di una CNN

Una CNN tipica è composta da una sequenza di layer che alternano operazioni di convoluzione, attivazione non lineare (tipicamente ReLU), e pooling, seguite da layer fully-connected finali per la classificazione.

Struttura generale. Un'architettura CNN standard segue il pattern [19], [30]:

$$\text{INPUT} \rightarrow [\text{CONV} \rightarrow \text{ReLU} \rightarrow \text{POOL}]^* \rightarrow [\text{FC} \rightarrow \text{ReLU}]^* \rightarrow \text{OUTPUT}$$

dove il blocco [CONV → ReLU → POOL] viene ripetuto più volte. Questa architettura stratificata, introdotta da [60] e consolidata da AlexNet [61], è diventata il template standard per le CNN moderne. In architetture rappresentative, come ad esempio VGGNet [63], si osserva una progressione caratteristica:

- Le dimensioni spaziali diminuiscono gradualmente (ad esempio, ad esempio, da un input di 224×224 pixel a feature map di 7×7 negli strati finali)
- Il numero di canali (feature map) aumenta gradualmente (dai 3 canali RGB iniziali a 512 o più nei layer profondi)
- I layer iniziali catturano feature di basso livello (bordi, colori, texture semplici)
- I layer profondi catturano feature di alto livello sempre più astratte (parti di oggetti, pattern complessi)

Questa progressione gerarchica riflette il principio ispiratore delle CNN: imitare la percezione visiva biologica, costruendo rappresentazioni complesse a partire da elementi semplici e locali.

Evoluzione storica. LeNet-5 [60], una delle prime CNN di successo, fu progettata per il riconoscimento di cifre manoscritte e conteneva 2 layer convoluzionali seguiti da layer fully-connected (Figura 2.8).

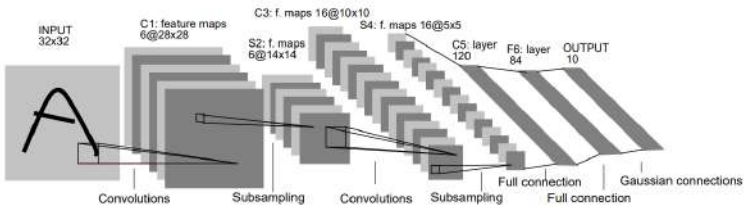

Figure 2.8: Architettura di LeNet-5, una delle prime reti convoluzionali di successo. La rete processa immagini 32×32 attraverso layer convoluzionali alternati a subsampling (pooling), seguiti da layer fully-connected. Fonte: [60].

AlexNet [61], che vinse la competizione ImageNet nel 2012 con grande margine, segnò l'inizio dell'era moderna del deep learning per la visione artificiale (Figura 2.9). L'architettura introdusse l'uso intensivo di ReLU, dropout per regolarizzazione, e training su GPU, stabilendo il template per le CNN moderne.

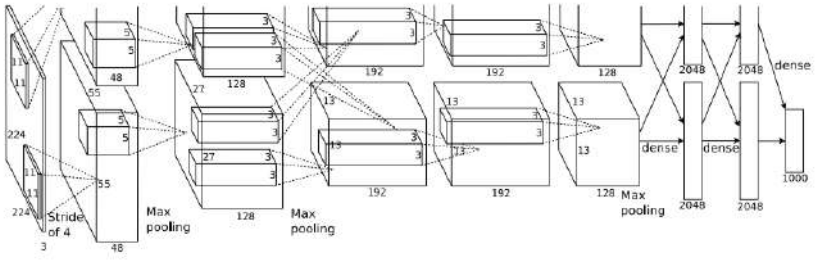

Figure 2.9: Architettura di AlexNet. La rete processa immagini 224×224×3 attraverso 5 layer convoluzionali (con max pooling) seguiti da 3 layer fully-connected, per un totale di circa 60 milioni di parametri. L'architettura è divisa su due GPU (rami superiore e inferiore). Fonte: [61].

Successivamente, architetture come VGGNet [63] dimostrarono l'efficacia di reti molto profonde ($16 - 19$ layer) con filtri piccoli (3×3), mentre ResNet [58] introdusse le **residual connections** (skip connections) che permettono di addestrare reti estremamente profonde (fino a 152 layer) mitigando il problema del vanishing gradient. Le skip connections permettono al gradiente di fluire direttamente attraverso la rete, facilitando l'ottimizzazione di architetture molto profonde.

Transfer Learning

Il **transfer learning** è una tecnica fondamentale nelle applicazioni pratiche di CNN e riveste particolare importanza per l'anomaly detection industriale, dove i dati di training sono tipicamente limitati e costosi da acquisire [64].

Principio. Invece di addestrare una CNN da zero con inizializzazione casuale, si parte da una rete pre-addestrata su un dataset molto grande (tipicamente ImageNet, con 1.4 milioni di immagini) e si adatta al nuovo task specifico.

Questo funziona perché i layer convoluzionali apprendono feature gerarchiche trasferibili: i layer iniziali rilevano pattern generici (bordi, texture, colori) utili per qualsiasi task di visione, mentre i layer profondi catturano rappresentazioni più astratte che possono essere adattate a nuovi domini [65].

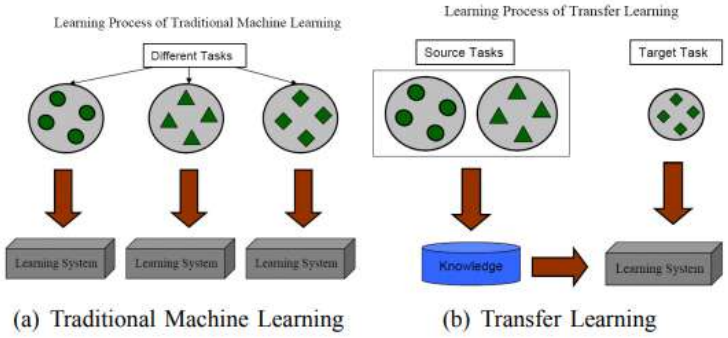

Figure 2.10: Confronto tra apprendimento tradizionale (a) e transfer learning (b). Nell'apprendimento tradizionale, ogni task è appreso indipendentemente; nel transfer learning, la conoscenza da task sorgente viene trasferita al task target. Fonte: [64].

Modalità di utilizzo. Esistono due approcci principali:

1. Feature Extraction: si congelano i layer convoluzionali pre-addestrati e si addestra solo un nuovo classificatore finale. Questo sfrutta le rappresentazioni apprese come feature general-purpose.

2. Fine-Tuning: dopo il feature extraction, si scongelano alcuni layer convoluzionali e si continua l'addestramento con learning rate molto basso. I layer profondi si adattano al task specifico, mentre quelli iniziali (feature generiche) cambiano poco. La scelta di quali layer scongelare dipende dalla dimensione del dataset: con pochi dati, si congelano più layer per evitare overfitting.

Vantaggi per l'anomaly detection. Nel contesto dell'anomaly detection industriale, il transfer learning offre vantaggi cruciali:

- **Migliori prestazioni con dati limitati**: le CNN pre-addestrate permettono di ottenere risultati competitivi anche con poche centinaia o migliaia di immagini, quantità tipiche nei dataset industriali

- **Convergenza più rapida**: partendo da rappresentazioni già significative, il training converge più velocemente e richiede meno epoch
- **Trasferibilità cross-domain**: anche quando le immagini industriali differiscono significativamente dalle immagini naturali di ImageNet, le feature di basso livello (bordi, texture) rimangono utili e trasferibili

Studi empirici su dataset di anomaly detection industriale, come MVTec AD [66], hanno dimostrato che modelli basati su transfer learning da ImageNet superano sistematicamente approcci addestrati da zero, confermando l'efficacia di questa strategia anche in domini visivamente distanti dalle immagini naturali.

In sintesi, il transfer learning permette di sfruttare la vasta conoscenza visiva codificata in reti addestrate su milioni di immagini, adattandola con costi computazionali e di annotazione ridotti a task specializzati come il rilevamento di anomalie in contesti industriali. Questa tecnica è diventata la pratica standard nell'anomaly detection visiva e costituirà un componente fondamentale dei metodi che esamineremo nel capitolo successivo.

Chapter 3

Anomalie

3.1 Anomaly Detection

Negli ultimi anni si è registrato un crescente interesse verso l'applicazione di tecniche di deep learning al rilevamento di anomalie per garantire la qualità dei prodotti e migliorare l'efficienza della produzione industriale [67]. L'anomaly detection (rilevamento di anomalie) è un task fondamentale nel machine learning che consiste nell'identificare pattern nei dati che deviano significativamente dal comportamento atteso o normale [68].

Un'**anomalia** viene definita come un'osservazione (o un sottoinsieme di osservazioni) apparentemente incoerente rispetto al resto dei dati [69]. A seconda del contesto applicativo, tali osservazioni possono essere indicate come outlier, novelty, o con denominazioni quali oggetto insolito, irregolare, atipico, raro, difettoso o fraudolento [70]. L'anomaly detection trova oggi impiego in una vasta gamma di settori, dalla cybersecurity [71] alle scienze della terra [72], in ambiti finanziari [73], per la diagnosi medica [74] e nella bioinformatica [75].

3.1.1 Definizione e Contesto Applicativo

Formalmente, dato un insieme di osservazioni $\mathcal{D} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$, l'obiettivo dell'anomaly detection è identificare un sottoinsieme $\mathcal{D}_{anomaly} \subset \mathcal{D}$ i cui elementi sono statisticamente diversi dalla maggioranza dei dati [68].

A differenza della classificazione tradizionale, dove si dispone di esempi etichettati per ciascuna classe, nell'anomaly detection si assume tipicamente la disponibilità di soli campioni normali durante la fase di training [76]. La natura prevalentemente non supervisionata del problema rende meno immediata la definizione di obiettivi di representation learning adeguati. L'approccio comunemente adottato consiste quindi nell'apprendere un modello dei campioni normali e considerare anomale le deviazioni rispetto ad esso.

Questo approccio si distingue dalla classificazione supervisionata per diverse caratteristiche fondamentali:

- **Classificazione one-class**: durante il training, il modello ha accesso principalmente (o esclusivamente) a esempi della classe normale. Le anomalie sono assenti, rare o troppo diverse tra loro per essere modellate efficacemente [76], [77].
- **Natura sbilanciata e variabilità**: nei contesti reali, le anomalie costituiscono una frazione minima dei dati (spesso $< 1\%$), [78] e possono manifestarsi in forme altamente variabili e imprevedibili [66].

Anomaly Detection nell'Industria. Nel contesto industriale, l'anomaly detection riveste un ruolo cruciale per il controllo qualità automatico e l'ispezione visiva [79]. Tradizionalmente, l'ispezione di prodotti richiedeva intervento manuale, risultando costosa, lenta e difficilmente scalabile a produzioni ad alto volume.

I metodi di anomaly detection basati su deep learning promettono di automatizzare questo processo, consentendo ispezione in tempo reale, consistente e scalabile. Tuttavia, il rilevamento di anomalie visive in ambito industriale affronta diverse sfide pratiche intrinseche agli scenari reali [66], [67]:

- **Variabilità delle condizioni di produzione**: fattori ambientali, variazioni di illuminazione e angolazioni della telecamera influenzano significativamente l'aspetto visivo dei prodotti. I dati considerati normali possono presentare un'elevata variabilità intra-classe, richiedendo ai metodi di rilevamento di mantenere robustezza in condizioni operative diverse per evitare falsi positivi.
- **Forte sbilanciamento dei dati**: le istanze normali superano di gran lunga quelle anomale, che spesso non sono etichettate. Questo rende difficile sia la raccolta di un training set rappresentativo che la definizione esatta di cosa costituisca un'anomalia.
- **Eterogeneità delle anomalie**: i difetti industriali possono manifestarsi in forme altamente variabili (graffi, crepe, contaminazioni, disallineamenti logici, parti mancanti, deviazioni dimensionali). A differenza di applicazioni generiche, il contesto industriale richiede il rilevamento di anomalie con caratteristiche strutturali e semantiche specifiche, inclusi micro-difetti superficiali che possono essere estremamente sottili.
- **Rumore e artefatti**: imperfezioni dei sensori o condizioni ambientali difficili generano elevati livelli di rumore nelle immagini, che possono complicare ulteriormente il rilevamento e portare a falsi allarmi.
- **Necessità di localizzazione precisa**: è necessario non solo classificare un'immagine come anomala ma anche identificare precisamente la regione difettosa (segmentazione pixel-level) per guidare eventuali interventi correttivi.

Queste sfide motivano lo sviluppo di metodi specializzati che vadano oltre il rilevamento di anomalie generico, incorporando conoscenze specifiche del dominio industriale.

3.1.2 Approcci all'Anomaly Detection

La letteratura distingue tradizionalmente tre paradigmi principali per l'anomaly detection, basati sulla disponibilità e natura dei dati di training [70], [80].

Supervised Anomaly Detection. Richiede un dataset di training completo, contenente esempi etichettati sia di campioni normali che anomali. Sebbene concettualmente più semplice, questo approccio è raramente applicabile in contesti industriali per due ragioni principali: (1) la difficoltà di raccogliere campioni rappresentativi di tutte le possibili tipologie di difetti, e (2) il fatto che le anomalie sono per natura rare e imprevedibili, rendendo impossibile definire a priori tutte le loro manifestazioni [79].

Semi-Supervised Anomaly Detection (One-Class Learning). L'approccio predominante nell'ispezione industriale moderna [66], [81], [82]. Il modello viene addestrato esclusivamente su immagini normali (anomaly-free) durante la fase di training, senza mai osservare esempi di anomalie. Al tempo di test, il sistema deve identificare le deviazioni dalla distribuzione normale appresa

Questo paradigma, noto anche come **one-class classification** [77] o **novelty detection**, riflette realisticamente gli scenari produttivi dove:

- I campioni difettosi sono rari, difficili da raccogliere ed estremamente variabili
- Si può garantire la qualità dei dati di training (solo prodotti certificati come non difettosi)
- L'obiettivo è apprendere una rappresentazione compatta della "normalità" e rilevare qualsiasi deviazione

La maggior parte dei metodi deep learning per anomaly detection industriale segue questo paradigma, sfruttando reti pre-addestrate e tecniche di transfer learning per apprendere robuste rappresentazioni della classe normale.

Fully Unsupervised Anomaly Detection. In questo scenario meno comune, si assume che l'insieme di training contenga una frazione sconosciuta di anomalie mescolate ai campioni normali, senza alcuna informazione di etichetta [68]. L'obiettivo è identificare i campioni che deviano dalla distribuzione maggioritaria utilizzando tecniche come clustering, stima di densità, o autoencoder. Questo approccio è raro nell'ispezione industriale, dove tipicamente si ha controllo sulla qualità dei dati di training e si garantisce che contengano solo campioni normali.

3.2 Evoluzione dei Metodi di Anomaly Detection

Lo sviluppo di metodi di deep learning per l'anomaly detection ha seguito un'evoluzione rapida negli ultimi anni. Presentiamo qui i principali approcci in ordine cronologico, con attenzione ai metodi moderni che rappresentano lo stato dell'arte attuale.

3.2.1 Era dei Metodi GAN-based (2017-2019)

I primi tentativi di applicare il deep learning all'anomaly detection sfruttarono le Generative Adversarial Networks (GAN) [83]. L'idea fondamentale era addestrare un generatore a sintetizzare immagini realistiche utilizzando esclusivamente campioni normali, per poi rilevare anomalie misurando la difficoltà di ricostruzione o la distanza dalla distribuzione appresa.

AnoGAN. AnoGAN [84] (Figura 3.1) è stato tra i primi, introducendo un approccio basato su mapping nello spazio latente: data un'immagine di test, si cerca iterativamente il vettore latente $\mathbf{z}^*$ che genera l'immagine più simile, combinando *residual loss* (distanza pixel-wise) e *discrimination loss* (similarità nelle feature del discriminatore). L'anomaly score finale è dato dalla loss ottimizzata. Limitazione principale: ottimizzazione iterativa lenta al test time.

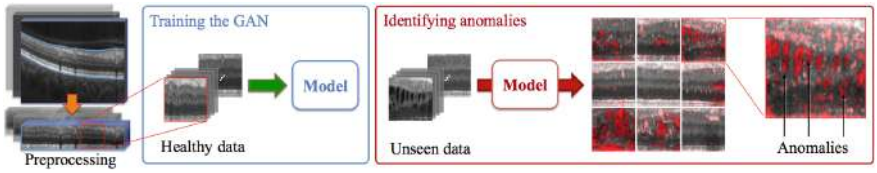

Figure 3.1: Framework di anomaly detection di AnoGAN. La fase di preprocessing include l'estrazione e l'appiattimento dell'area, l'estrazione di patch e la normalizzazione dell'intensità. L'addestramento generativo avversario (GAN) viene eseguito su dati sani, mentre il testing viene effettuato sia su casi sani non visti che su dati anomali. Fonte: [84]

GANomaly. Per ovviare a questa inefficenza, GANomaly [85] introdusse un'architettura encoder-decoder-encoder più efficiente, eliminando l'ottimizzazione iterativa. L'anomaly score si basava sulla distanza tra rappresentazioni latenti prima e dopo la ricostruzione (Figura 3.2).

f-AnoGAN e Skip-GANomaly. f-AnoGAN [86] risolse ulteriormente il problema di efficienza addestrando un encoder che mappava direttamente immagini nel latent space (Figura 3.3), riducendo i tempi di inferenza da minuti a millisecondi.

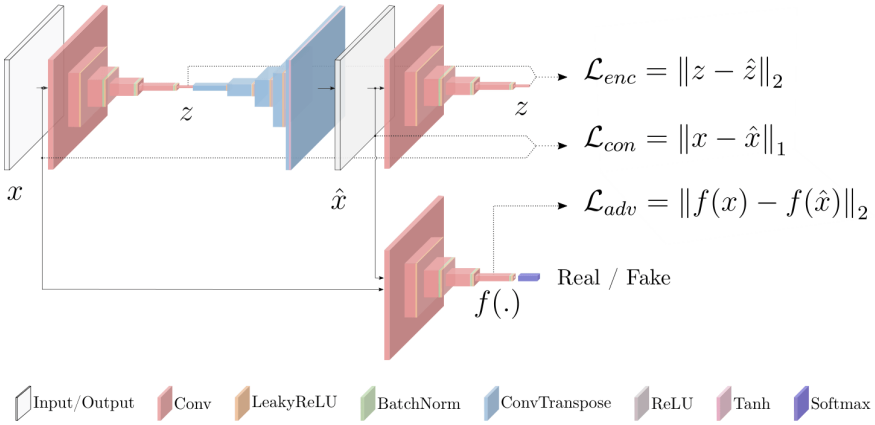

Figure 3.2: Architettura di GANomaly. Il sistema utilizza una struttura encoder-decoder-encoder dove l'immagine di input x viene codificata in uno spazio latente $\tilde{z}$, ricostruita come $\hat{x}$, e poi nuovamente codificata. L'anomaly score si basa sulla distanza tra le rappresentazioni latenti ($\mathcal{L}_{enc}$), sulla ricostruzione dell'immagine ($\mathcal{L}_{con}$) e sulle feature del discriminatore ($\mathcal{L}_{adv}$). Fonte: [85]

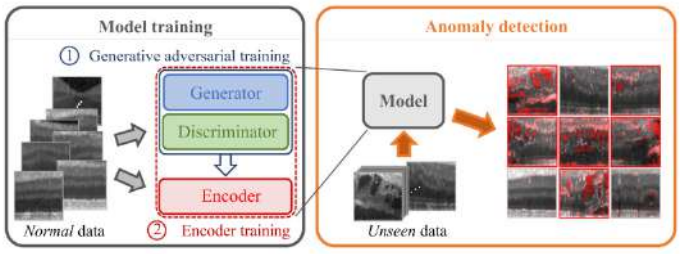

Figure 3.3: Framework di f-AnoGAN. Il training del modello avviene in due fasi: addestramento GAN (generator e discriminator) e addestramento dell'encoder (componente chiave che elimina l'ottimizzazione iterativa di AnoGAN). Entrambe le fasi utilizzano solo dati normali. L'anomaly detection viene poi eseguita su dati *unseen*, identificando regioni anomale evidenziate in rosso. Adattata da [86].

Skip-GANomaly [87] migliorò la qualità delle ricostruzioni aggiungendo skip connections U-Net-style per preservare dettagli spaziali (Figura 3.4).

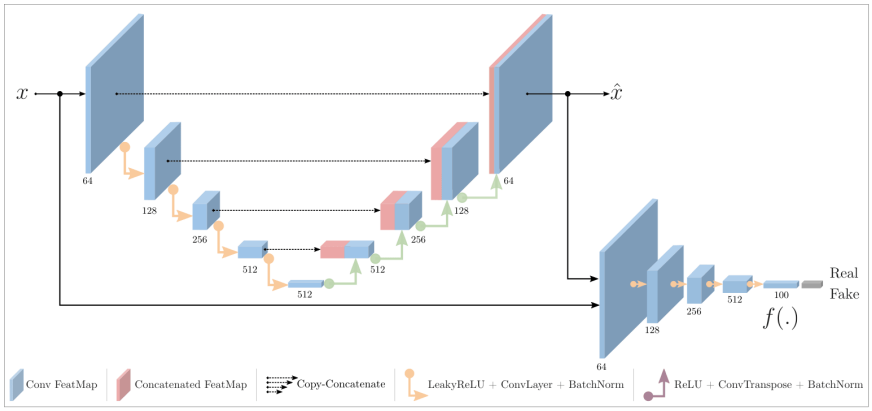

Figure 3.4: Architettura di Skip-GANomaly. Il sistema estende GANomaly aggiungendo skip connections in stile U-Net tra encoder e decoder, preservando informazioni spaziali ad alta risoluzione a diversi livelli. Questo migliora la qualità delle ricostruzioni e la capacità di localizzazione delle anomalie. Fonte: [87]

Limitazioni dei metodi GAN. Sebbene pionieristici, i metodi GAN-based sono stati progressivamente abbandonati per diverse ragioni: (1) training instabile tipico delle GAN, (2) difficoltà di convergenza e bilanciamento delle loss multiple , (3) performance inferiori ai metodi successivi più semplici. I benchmark recenti dimostrano che approcci più semplici superano consistentemente i metodi GAN [6], [81].

3.2.2 Autoencoder-based Methods (2016-2021)

Gli autoencoder rappresentano un paradigma intuitivo per l'anomaly detection [80]: addestrati a ricostruire immagini normali minimizzando l'errore $\mathcal{L} = \|\mathbf{x} - D(E(\mathbf{x}))\|^2$, si ipotizza che falliscano nel ricostruire accuratamente anomalie mai viste, poiché queste ultime si discostano dai pattern appresi. Di conseguenza, un valore elevato della loss di ricostruzione su un'immagine di test segnala una potenziale anomalia. L'anomaly score è dato appunto dall'errore di ricostruzione. Questo approccio dominò la letteratura tra il 2016 e il 2020, con diverse varianti.

Autoencoder Deterministici. Gli Autoencoder Deterministici utilizzano encoder-decoder standard con funzioni di perdita basate sull'errore di ricostruzione, come l'Errore Quadratico Medio (MSE) o l'indice SSIM (Structural Similarity Index Measure). Il benchmark MVTec AD [66] valutò autoencoder con entrambe le loss, riscontrando in tutti i casi performance modeste rispetto agli standard attuali.

Variational Autoencoder (VAE). Variational Autoencoder (VAE) [88], [89] introducono un framework probabilistico, modellando il latent space come distribuzione Gaussiana. Il VAE viene addestrato massimizzando l'ELBO (Evidence Lower BOund), equivalente a minimizzare la seguente loss:

$$\mathcal{L}_{VAE} = \underbrace{\|\mathbf{x} - D(\mathbf{z})\|^2}_{\text{Reconstruction}} + \underbrace{\text{KL}(q(\mathbf{z}|\mathbf{x})\|p(\mathbf{z}))}_{\text{Regularization}} \tag{3.1}$$

dove il termine KL regolarizza il latent space verso una Gaussiana $p(\mathbf{z}) = \mathcal{N}(0, I)$. L'anomaly score combina errore di ricostruzione e distanza dalla prior. Tuttavia, le performance rimasero comparabili o inferiori agli AE standard per anomaly detection industriale.

Memory-Augmented Autoencoder. Per risolvere il problema del *reconstruction bias* (autoencoder che ricostruiscono bene anche anomalie), MemAE [90] introduce un modulo di memoria tra encoder e decoder (Figura 3.5): feature dell'encoder vengono recuperate dal memory module di prototipi normali tramite attention, forzando la ricostruzione attraverso pattern normali memorizzati. MNAD [91] estese questo approccio con predizione multi-frame. Entrambi furono valutati in MVTec LOCO AD [1] con risultati modesti.

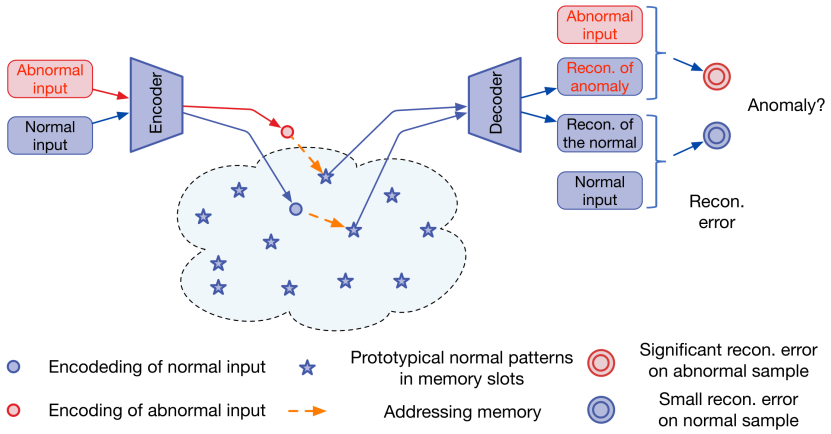

Figure 3.5: Anomaly detection tramite Memory-Augmented Autoencoder (MemAE). Dopo il training su un dataset contenente solo campioni normali, il modulo di memoria in MemAE registra prototipi di pattern normali. Dato un input anomalo, MemAE recupera i pattern normali più rilevanti in memoria per la ricostruzione, risultando in un output significativamente diverso dall'input anomalo. Gli input normali (blu) vengono mappati vicino ai prototipi in memoria (stelle blu), mentre gli input anomali (rosso) vengono forzatamente ricostruiti attraverso i pattern normali, producendo un errore di ricostruzione elevato. Fonte: [90]

Limitazioni e Declino. Nonostante l'intuizione semplice, gli autoencoder presentano limitazioni intrinseche che ne hanno causato l'abbandono:

- **Reconstruction Bias**: non ci sono garanzie che le anomalie siano ricostruite peggio. L'autoencoder impara rappresentazioni generiche che possono ricostruire anche pattern mai visti [66].
- **Information Loss**: il bottleneck comprime informazione, ma questa compressione non è discriminativa per anomalie. Dettagli cruciali possono essere persi, sia normali che anomali.
- **Training Difficulty**: bilanciare la capacità del modello è critico. Troppa capacità ricostruisce anche anomalie, mentre troppo poca produce falsi positivi su normali.
- **Performance Inferiori**: benchmark come MVTec AD e MVTec LOCO AD mostrarono che autoencoder venivano consistentemente superati da metodi feature-based e student-teacher.

Tra il 2020 e 2021, divenne chiaro che feature pre-addestrate da reti come ResNet catturavano pattern normali meglio di encoder addestrati da zero, e metodi non parametrici come k-NN evitavano il reconstruction bias. Questo portò al graduale abbandono degli autoencoder come metodo primario. Varianti moderne sopravvivono solo come componenti di architetture ibride (es. DRAEM, EfficientAD).

3.2.3 Primi Metodi Feature-based (2020-2021)

Un cambio di paradigma avvenne con metodi che sfruttarono feature estratte da reti pre-addestrate, evitando training di generatori complessi. Questi metodi condividevano l'intuizione che le feature estratte da reti pre-addestrate contenessero informazioni sufficienti per modellare la normalità, senza necessità di complessi meccanismi generativi. La differenza principale risiedeva in *come* queste feature venivano modellate: PaDiM utilizzava distribuzioni parametriche (Gaussiane), Uninformed Students sfruttava un paradigma teacher-student, e SPADE adottava un approccio non-parametrico basato su nearest neighbors.

PaDiM (2020). PaDiM (Patch Distribution Modeling) [82] (Figura 3.6) modella la distribuzione normale tramite feature multi-scala da CNN preaddestrate (tipicamente ResNet/WideResNet su ImageNet). Per ogni posizione spaziale (i, j), concatena feature da layer a diverse profondità:

$$x_{ij} = [\mathbf{f}_1^{(i,j)}, \mathbf{f}_2^{(i,j)}, \mathbf{f}_3^{(i,j)}] \tag{3.2}$$

Per ogni posizione, modella la distribuzione come Gaussiana multivariata $\mathcal{N}(\mu_{ij}, \Sigma_{ij})$ stimata dal training set. Al test, l'anomaly score è dato dalla distanza di Mahalanobis:

$$M_{ij} = \sqrt{(x_{ij} - \mu_{ij})^T \Sigma_{ij}^{-1} (x_{ij} - \mu_{ij})} \tag{3.3}$$

PaDiM applica random dimensionality reduction per efficienza. Vantaggi: no training, localizzazione precisa, robustezza. Limitazioni: memoria per matrici di covarianza.

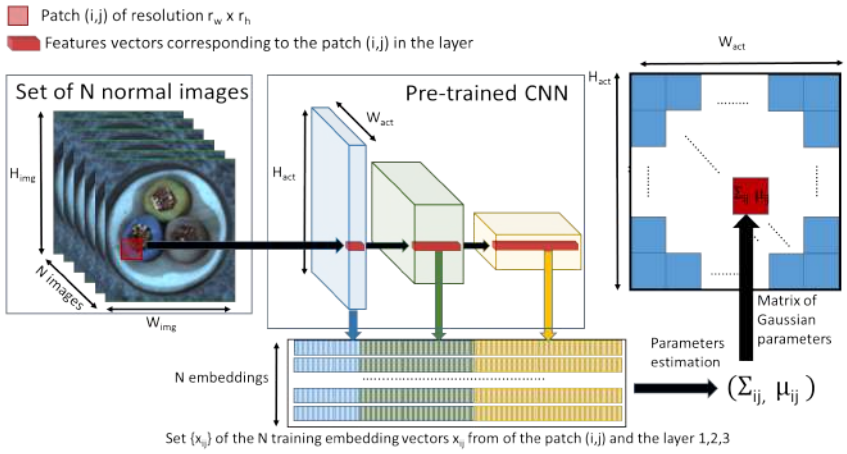

Figure 3.6: Funzionamento di PaDiM. Per ogni posizione (i, j) nella feature map, PaDiM concatena feature estratte da tre layer CNN pre-addestrati a diverse profondità e apprende i parametri di una distribuzione Gaussiana (μ_{ij}, Σ_{ij}) dall'insieme di vettori di training.

Uninformed Students (2020). Bergmann et al. [79] introdussero un approccio student-teacher (Figura 3.7): un teacher network pre-addestrato fornisce supervisione a un insieme di student networks addestrati solo su dati normali. Gli studenti imparano a replicare l'output del teacher per immagini normali ma non sono in grado di farlo per anomalie non viste durante l'addestramento. L'anomaly score misura la discrepanza tra l'output del teacher e la media degli output degli studenti: regioni con alta discrepanza indicano potenziali anomalie.

SPADE (2021). SPADE (Sub-image Anomaly Detection) [92] utilizza k-nearest neighbors (k-NN) nello spazio delle feature profonde. Costruisce una gallery $\mathcal{G}$ memorizzando tutte le feature di tutte le posizioni spaziali delle immagini normali. Al test, per ogni posizione p, recupera i κ nearest neighbors dalla gallery e calcola:

$$d(\mathbf{y}, p) = \frac{1}{\kappa} \sum_{f \in \mathcal{N}_\kappa(F(\mathbf{y},p))} \|f - F(\mathbf{y}, p)\|_2 \tag{3.4}$$

Per gestire efficientemente la gallery — che può contenere milioni di feature — SPADE utilizza strutture dati ottimizzate per la ricerca approssimata dei vicini più prossimi (Approximate Nearest Neighbor search).

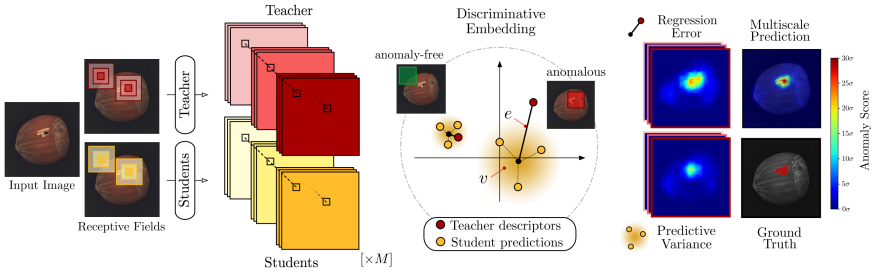

Figure 3.7: Schema dell'approccio Uninformed Students. Le immagini di input vengono processate da una rete teacher che estrae feature per regioni locali (receptive fields). Un ensemble di M student networks viene addestrato a predire l'output del teacher solo su dati anomaly-free. Durante l'inferenza, gli students producono errori di regressione e e incertezze predittive v elevati per i pixel in cui il receptive field copre regioni anomale. Le mappe di anomalia generate con receptive fields di diverse dimensioni possono essere combinate per segmentazione multi-scala. Fonte: [79]

3.2.4 Normalizing Flow-based Methods (2021-2022)

Un'ulteriore evoluzione nei metodi di anomaly detection fu l'introduzione dei normalizing flows, modelli probabilistici in grado di modellare esplicitamente distribuzioni complesse. A differenza dei metodi precedenti che si basavano su distanze o confronti diretti, i flow permettono di calcolare esattamente la probabilità (likelihood) dei dati, offrendo una modellazione più espressiva della normalità

CFLOW-AD (2021). CFLOW-AD [93] usa conditional normalizing flows per modellare la distribuzione delle feature estratte da reti pre-addestrate (Figura 3.8). Il flow trasforma la distribuzione complessa delle feature normali in una Gaussiana standard, permettendo di calcolare direttamente la likelihood. L'anomaly score è dato dalla negative log-likelihood: campioni anomali, avendo bassa probabilità sotto il modello appreso, ricevono punteggi elevati.

FastFlow (2021). FastFlow [94] introduce un'architettura 2D normalizing flow estremamente efficiente basata su 2D coupling layers, preservando la struttura spaziale delle feature (Figura 3.9). A differenza di CFLOW-AD che opera su vettori appiattiti di feature, FastFlow mantiene la struttura 2D, risultando più veloce e accurato. Il metodo raggiunse performance competitive con tempi di inferenza ridotti, rendendo i flow pratici per applicazioni in tempo reale.

CS-Flow (2022). CS-Flow (Cross-Scale Flow) [95] estende l'approccio flow-based a feature multi-scala (Figura 3.10). Utilizza normalizing flows separati su feature estratte da layer diversi di una rete pre-addestrata, catturando anomalie sia a livello locale (texture, dettagli) che globale (struttura, forma). L'anomaly

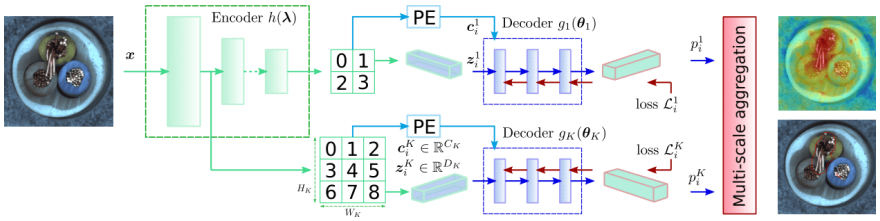

Figure 3.8: Architettura di CFLOW-AD. L'encoder CNN con pyramid pooling estrae feature multi-scala che catturano informazioni sia globali che locali. Per ogni scala k, un decoder basato su conditional normalizing flow processa le feature z_i^k con input condizionale c_i^k contenente informazioni spaziali da un positional encoder (PE). Le likelihood multi-scala p_i^k vengono aggregate per produrre la mappa di anomalia finale. Fonte: [93]

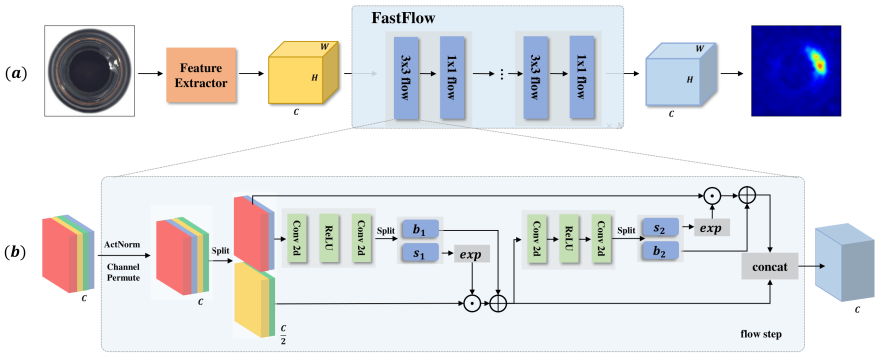

Figure 3.9: Pipeline di FastFlow per anomaly detection. (a) L'architettura completa comprende un feature extractor (CNN o vision transformer) seguito dal modello FastFlow, composto da flow layers alternati "3×3" e "1×1". (b) Dettaglio di un flow step: le feature vengono processate attraverso operazioni di convoluzione, split, attivazioni esponenziali e concatenazione per preservare la struttura spaziale 2D. Fonte: [94]

score aggrega likelihood da tutte le scale. CS-Flow è ancora competitivo nei benchmark recenti.

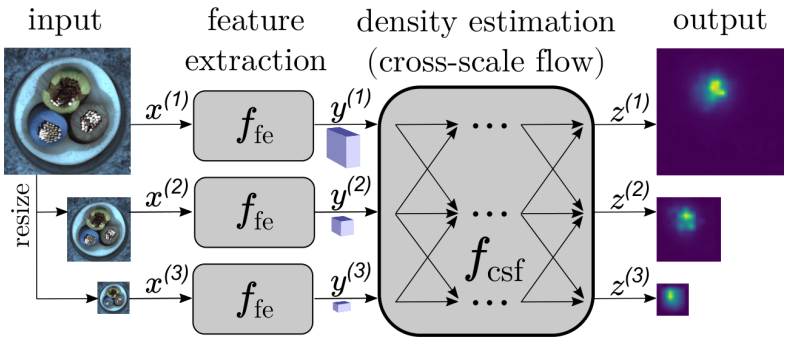

Figure 3.10: Schema di CS-Flow. Il metodo processa feature estratte da input multi-scala attraverso normalizing flows con cross-connections tra scale (f_{csf}). Ogni scala elabora separatamente le feature (f_{fe}) per catturare anomalie a diversi livelli di granularità, dalla texture fine alla struttura globale. Fonte: [95]

3.2.5 Synthetic Anomaly Generation (2021)

Un approccio innovativo alla sfida della scarsità di dati anomali consiste nel generare sinteticamente anomalie durante il training, trasformando il problema da puramente unsupervised in una forma di apprendimento supervisionato con etichette generate automaticamente. L'idea fondamentale è che, sebbene non si disponga di esempi reali di difetti, sia possibile creare simulazioni plausibili che insegnino al modello a distinguere tra normale e anomalo.

CutPaste (2021). CutPaste [96] adotta una strategia molto semplice (Figura 3.11): genera anomalie sintetiche tagliando una patch rettangolare da un'immagine normale e incollandola in una posizione casuale della stessa immagine, eventualmente applicando trasformazioni aggiuntive (rotazione, variazione di colore). Una CNN viene poi addestrata come classificatore binario a distinguere tra immagini originali (classe normale) e immagini manipolate (classe anomalia sintetica). Vantaggi principali: implementazione semplice, training veloce, nessuna dipendenza da feature pre-addestrate. Tuttavia, le anomalie generate sono limitate a difetti di tipo "oggetto dislocato" e non generalizzano a tutti i tipi di difetti.

DRAEM (2021). DRAEM (Discriminatively Trained Reconstruction Embedding) [97] combina generazione di anomalie realistiche con un meccanismo di ricostruzione. Utilizza texture di difetti reali provenienti da dataset esterni per creare anomalie visivamente plausibili (Figura 3.12). L'architettura comprende:

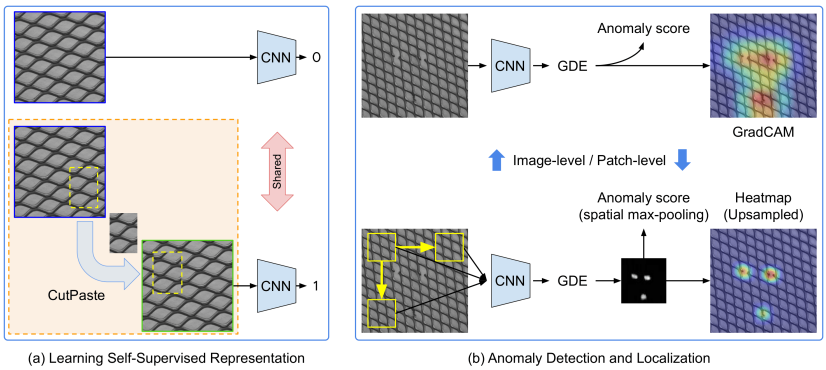

Figure 3.11: Overview di CutPaste. (a) Training: una CNN viene addestrata a distinguere immagini normali (blu) da immagini augmented (verde) tramite *CutPaste* (riquadro arancione), che taglia una regione rettangolare (riquadro giallo) e la incolla in posizione casuale. Le rappresentazioni possono essere apprese a livello di immagine intera o patch locali. (b) Detection e localizzazione: la rappresentazione image-level produce un anomaly score globale e localizza difetti via GradCAM; la rappresentazione patch-level estrae feature dense per produrre mappe di anomalia dettagliate. Fonte: [96]

- Un network di ricostruzione (tipicamente un autoencoder) addestrato a ripristinare l'immagine originale dall'input alterato
- Un discriminatore che localizza pixel-level le regioni anomale (differenza tra input e output)

Durante il training, il ricostruttore impara a ricostruire le anomalie sintetiche, mentre il discriminatore impara a identificarle. Al test time, le deviazioni nella ricostruzione e l'output del discriminatore vengono combinati per produrre mappe di anomalia dettagliate. DRAEM eccelle nella localizzazione precisa dei difetti e può gestire un'ampia varietà di texture anomale, ma richiede l'accesso a dataset di texture esterne e un training più complesso.

3.2.6 Memory-based Methods (2022)

Un'evoluzione dei metodi feature-based è rappresentata dai **memory-based methods**, che esplicitano e ottimizzano il concetto di "memoria" dei pattern normali. Mentre approcci come PaDiM modellano statisticamente la distribuzione normale e SPADE utilizza l'intero training set come gallery, i memory-based methods costruiscono attivamente una *memory bank* di prototipi rappresentativi. Questo supera le limitazioni di approcci precedenti: metodi non parametrici (SPADE) soffrono di elevato uso di memoria e tempi di ricerca, mentre metodi parametrici (autoencoders, GAN) tendono a generalizzare eccessivamente, perdendo la capacità di rilevare anomalie sottili.

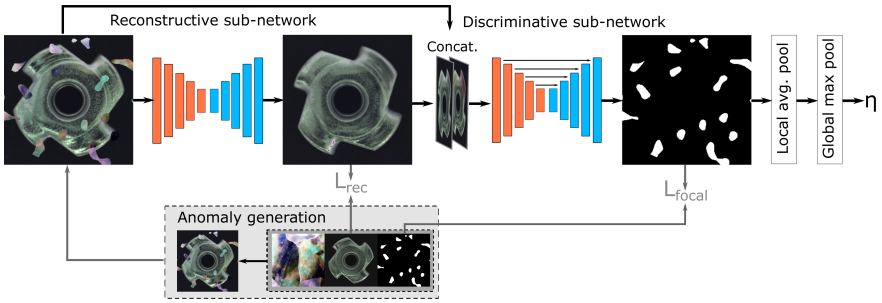

Figure 3.12: Processo di anomaly detection di DRAEM. Le regioni anomale vengono implicitamente rilevate e corrette (inpainting) dalla sotto-rete ricostruttiva addestrata con $\mathcal{L}_{rec}$. L'output della sotto-rete e l'immagine di input vengono concatenati e processati dalla sotto-rete discriminativa. La rete di segmentazione, addestrata utilizzando la Focal loss $\mathcal{L}_{focal}$, localizza le regioni anomale producendo una mappa di anomalia. L'anomaly score η a livello immagine è derivato dalla mappa. Fonte: [97]

PatchCore (2022). PatchCore [81] è rapidamente divenuto uno dei baseline di riferimento per l'anomaly detection industriale, combinando accuratezza ed efficienza operativa e rimanendo competitivo anche nei benchmark più recenti (Figura 3.13).

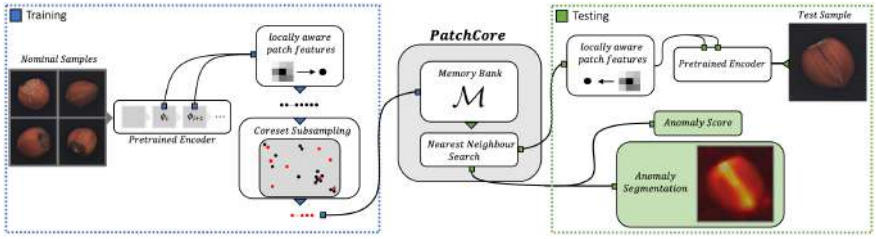

Figure 3.13: Overview di PatchCore. Durante il training (blu), i campioni normali vengono scomposti in una memory bank di feature a livello di patch neighbourhood-aware. Per ridurre ridondanza e tempo di inferenza, la memory bank viene compressa tramite un coreset subsampling greedy. Al momento del test (verde), le immagini sono classificate come anomale se almeno una patch è anomala, e viene generata una mappa di segmentazione dell'anomalia a livello di pixel valutando ciascuna patch-feature. Fonte: [81]

Locally Aware Patch Features. Per rilevare difetti sottili (micro-graffi, piccole contaminazioni, variazioni texture) è necessario preservare informazione spaziale ad alta risoluzione. Il metodo estrae **locally aware patch features** da layer intermedi di una rete convoluzionale pre-addestrata su ImageNet, tipicamente WideResNet50.

PatchCore estrae activation maps da due layer a risoluzioni diverse:

- **Layer 2**: risoluzione maggiore, cattura pattern locali e texture dettagliate

- **Layer 3**: risoluzione intermedia, codifica informazioni semantiche più astratte

Per ogni posizione spaziale (i, j), le feature dai due layer vengono concatenate e aggregate attraverso adaptive average pooling, per catturare sia anomalie a grana fine (layer superficiale) che deviazioni strutturali (layer profondo).

Preservando la risoluzione spaziale, tuttavia, ogni immagine di training genera centinaia o migliaia di patch features.

Coreset Subsampling. La soluzione innovativa di PatchCore è l'uso del **coreset subsampling**, un algoritmo greedy che seleziona un sottoinsieme rappresentativo minimizzando la massima distanza da qualsiasi punto al coreset.

Formalmente, dato l'insieme completo di patch features $\mathcal{M} = \{m_1, \ldots, m_N\}$ estratte dal training set, l'algoritmo costruisce iterativamente una memory bank $\mathcal{M}_\mathcal{C} \subset \mathcal{M}$ selezionando iterativamente patch fino a raggiungere una dimensione target N, tipicamente stabilita come una piccola percentuale del memory bank originale M.

Algorithm 1 Greedy Minimax Facility Location Coreset Selection for PatchCore

Require: Memory bank originale M (insieme di patch-feature), dimensione target del coreset N
Ensure: Coreset ottimizzato $M_C \subset M$
1: Inizializza $M_C = \{m_{\text{random}}\}$ con una patch-feature casuale da M
2: **while** $|M_C| < N$ **do**
3: Per ogni $m \in M \setminus M_C$, calcola la distanza dal vicino più prossimo in M_C:
4: $d(m, M_C) = \min_{n \in M_C} \|m - n\|_2$
5: Trova la patch-feature m^* più distante:
6: $m^* = \arg \max_{m \in M \setminus M_C} d(m, M_C)$
7: Aggiorna il coreset: $M_C \leftarrow M_C \cup \{m^*\}$
8: **end while**
9: **return** M_C

Questo algoritmo garantisce che la memory bank copra efficacemente lo spazio delle feature normali: ogni iterazione aggiunge il punto che migliora maggiormente la copertura delle regioni sotto-rappresentate. La dimensione tipica della memory bank è l'$1 - 10\%$ delle feature originali (es. $2,500 - 25,000$ patches invece di $250,000$), riducendo drasticamente memoria e tempi di ricerca mantenendo accuratezza elevata.

Anomaly Scoring e Re-weighting. Al test time, data un'immagine di test $\mathbf{x}_{\text{test}}$, PatchCore estrae l'insieme di patch-feature $P(\mathbf{x}_{\text{test}}) = P_{s,p}(\phi_j(\mathbf{x}_{\text{test}}))$. Per ogni patch-feature di test $m_{\text{test}} \in P(\mathbf{x}_{\text{test}})$, si calcola la distanza dal suo vicino più prossimo (nearest neighbour) nella memory bank M:

$$d(m_{\text{test}}) = \min_{m \in M} \|m_{\text{test}} - m\|_2. \tag{3.5}$$

Lo **score di anomalia a livello patch** è quindi $d(m_{\text{test}})$. Per ottenere uno **score di anomalia a livello immagine** robusto, prima si identifica prima la coppia critica (m^*_{test}, m^*) che massimizza la distanza:

$$m^*_{\text{test}}, m^* = \arg \max_{m_{\text{test}} \in P(\mathbf{x}_{\text{test}})} \arg \min_{m \in M} \|m_{\text{test}} - m\|_2, \tag{3.6}$$

e si definisce lo score grezzo $s^* = \|m^*_{\text{test}} - m^*\|_2$. Successivamente, viene applicato un **meccanismo di re-weighting** che tiene conto della rarità del vicino m^* all'interno della memory bank M. L'idea è che se m^* è esso stesso un pattern raro (cioè lontano dai suoi b vicini in M), allora la deviazione m^*_{test} è meno sorprendente e il punteggio viene attenuato. Lo score finale a livello immagine s è:

$$s = \left(1 - \frac{\exp(\|m^*_{\text{test}} - m^*\|_2)}{\sum_{m \in \text{Nb}(m^*)} \exp(\|m^*_{\text{test}} - m\|_2)}\right) \cdot s^*, \tag{3.7}$$

dove $\text{Nb}(m^*)$ denota l'insieme dei b patch-feature più vicini a m^* in M. La **mappa di segmentazione** (anomalia a livello pixel) viene ottenuta riallineando spazialmente gli score a livello patch $d(m_{\text{test}})$ per ogni posizione, seguita da un'upscaling tramite interpolazione bilineare e uno smoothing gaussiano ($\sigma = 4$) per rifinire il risultato.

Impatto e Limitazioni. PatchCore ha avuto un impatto significativo sulla ricerca e pratica industriale:

- **State-of-the-art su MVTec AD**: 99.1% image AUROC medio, 98.1% pixel AUROC. Baseline standard, utilizzato come confronto in praticamente tutti i paper successivi
- **Efficienza e semplicità**: inferenza rapida, nessun training richiesto, facile da implementare

Le limitazioni principali sono:

- **Memoria**: la memory bank, sebbene compressa, richiede comunque diversi GB per dataset grandi
- **Tempo di costruzione**: il coreset subsampling può richiedere ore per dataset molto ampi
- **Nessuna adattazione**: essendo non-parametrico, non può adattarsi a nuove distribuzioni senza ricostruire la memory bank

Nonostante queste limitazioni, PatchCore rimane uno dei metodi più pratici e affidabili, particolarmente quando l'accuratezza è prioritaria e il setup può essere fatto offline.

MemSeg (2022). MemSeg (Memory-guided Segmentation) [98] adotta un approccio diverso: invece di confrontare feature o ricostruire immagini, trasforma l'anomaly detection in un task di segmentazione semantica end-to-end (Figura 3.14). Sulla base dell'idea di una ridotta varianza intra-classe tra prodotti della stessa linea produttiva:

- introducendo anomalie simulate durante il training, il modello impara esplicitamente a discriminare normale da non-normale
- l'uso di un memory module consente di memorizzare pattern normali tipici per confrontare l'input

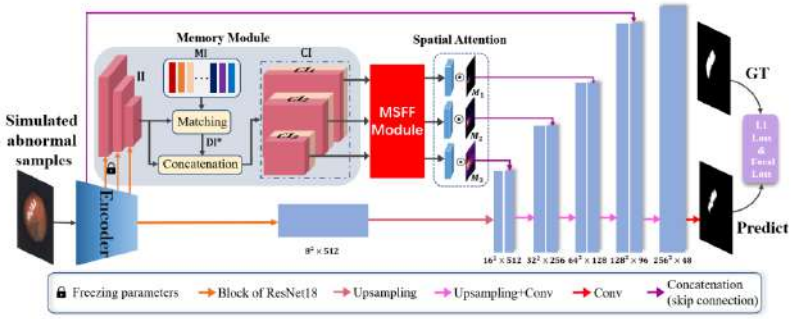

Figure 3.14: Architettura di MemSeg. Basato su architettura U-Net con encoder ResNet18 pre-addestrato, il metodo introduce anomalie simulate e un modulo di memoria per realizzare il rilevamento di difetti superficiali semi-supervisionato in maniera end-to-end. Inoltre, per fondere le informazioni della memoria con le feature di alto livello dell'immagine di input, è introdotto un modulo di fusione di feature multiscala (MSFF Module) e un modulo di attenzione spazialeFonte: [98]

Architettura U-Net Based. MemSeg usa U-Net [99] come architettura base. L'architettura si compone di:

1. **Encoder**: ResNet18 pre-addestrato su ImageNet che estrae feature a diverse risoluzioni
2. **Memory Module**: Memorizza feature di N campioni normali selezionati casualmente e calcola "difference information"
3. **MSFF Module**: Multi-Scale Feature Fusion module che fonde feature dell'input con difference information
4. **Decoder**: genera una mappa di segmentazione binaria (normale/anomalo) invece di ricostruire l'immagine

Memory Module e Difference Information. Si selezionano casualmente N immagini normali dal training set come "memory samples" e si estraggono feature che andranno a costituire la *memory information* MI.

Data un'immagine di input (normale o simulata anomala), si estraggono le sue feature II (*input information*) e si calcola la distanza L2 tra II e *ogni* memory sample MI_i:

$$DI_i = \|II - MI_i\|_2 \quad \text{per} \quad i = 1, \ldots, N \tag{3.8}$$

Questo produce N "difference information" $DI_1, \ldots, DI_N$. Si seleziona la *best difference information* DI^* come quella con somma minima su tutti gli elementi:

$$DI^* = \arg\min_{DI_i} \sum_{\text{elementi}} DI_i \tag{3.9}$$

Intuitivamente, DI^* rappresenta le differenze rispetto al memory sample più simile all'input.

Da DI^* si generano tre **spatial attention maps** M_1, M_2, M_3 che enfatizzano regioni potenzialmente anomale:

$$M_3 = \frac{1}{C_3} \sum_{i=1}^{C_3} DI^*_{3i} \tag{3.10}$$

$$M_2 = \frac{1}{C_2} \left(\sum_{i=1}^{C_2} DI^*_{2i} \right) \odot M_3^U \tag{3.11}$$

$$M_1 = \frac{1}{C_1} \left(\sum_{i=1}^{C_1} DI^*_{1i} \right) \odot M_2^U \tag{3.12}$$

dove C_n è il numero di canali di DI^*_n e M^U denota upsampling.

Multi-Scale Feature Fusion Module (MSFF). DI^* viene concatenato con II nella dimensione dei canali per ottenere la concatenated information CI_n ($n = 1, 2, 3$ per le tre risoluzioni). Queste contengono ridondanza (concatenazione di due sorgenti diverse) e vengono elaborate da MSFF

1. Convoluzione iniziale 3×3 che mantiene il numero di canali
2. Coordinate Attention (CA) [100] per catturare relazioni tra canali considerando informazioni spaziali
3. Multi-scale fusion: Feature da diverse risoluzioni vengono allineate (upsampling + conv) e sommate element-wise

Le feature fuse vengono poi pesate dalle spatial attention maps M_n e passate al decoder.

Loss Functions. La funzione di perdita complessiva combina L1 loss e Focal loss [101]:

$$\mathcal{L}_{l1} = \|S - P\|_1 \tag{8}$$

$$\mathcal{L}_{fl} = -(1 - p_t)^{\gamma} \log(p_t) \tag{9}$$

dove S è il ground truth delle regioni anomale nell'immagine simulata, P è il valore predetto dal modello, e p_t è definito come:

- $p_t = p$ quando il ground truth del pixel corrispondente in S è 1
- $p_t = 1 - p$ quando il ground truth del pixel in S è 0

con p la probabilità predetta della categoria del pixel.

La funzione obiettivo finale combina questi vincoli:

$$\mathcal{L} = \lambda_1 \cdot \mathcal{L}_{l1} + \lambda_2 \cdot \mathcal{L}_{fl} \tag{10}$$

dove λ_1 e λ_2 sono iperparametri di bilanciamento.

Vantaggi e Limitazioni. MemSeg offre vantaggi significativi rispetto ad approcci precedenti: la segmentazione end-to-end evita sia la ricostruzione (autoencoders/GAN) sia il feature matching complesso (embedding-based methods), risultando in velocità di inferenza molto superiori. Il modello si adatta alle specifiche anomalie del dominio, mentre la supervision diretta produce boundary di segmentazione molto accurati.

Tuttavia, il metodo presenta alcune limitazioni. MemSeg è sensibile all'allineamento spaziale degli oggetti: performance degradano quando gli oggetti nel test set hanno orientamenti casuali o posizioni molto variabili. A differenza di PatchCore, richiede un training completo del decoder e del MSFF module. Infine, la qualità della detection dipende fortemente da quanto le anomalie simulate durante training approssimano le anomalie reali, rendendo la strategia di simulation un componente critico del successo del metodo.

Confronto: PatchCore vs MemSeg. PatchCore e MemSeg rappresentano due filosofie complementari nell'uso della memoria per anomaly detection:

- **PatchCore**: Deployment rapido senza training, massima robustezza a variazioni spaziali, accuratezza eccellente. Ideale quando non si può/vuole addestrare e gli oggetti possono essere disallineati o ruotati.
- **MemSeg**: Velocità massima per real-time, segmentazione end-to-end, boundary precisi. Ideale per scenari industriali con oggetti spazialmente consistenti.

Entrambi i metodi superano significativamente approcci precedenti (PaDiM, SPADE) e hanno influenzato profondamente lo sviluppo successivo, stabilendo la memoria esplicita come componente chiave per l'anomaly detection ad alte prestazioni.

Table 3.1: Confronto tra approcci memory-based

Aspetto	PatchCore	MemSeg
Paradigma	Non-parametrico	Semantic segmentation
Training	No (solo coreset)	Sì (end-to-end)
Anomaly simulation	No	Sì
Memoria	Coreset di patch features	N immagini normali intere
Scoring	Nearest-neighbor distance	Segmentation diretta
Output	Distance map	Binary segmentation
Velocità inferenza	Veloce	Molto veloce
Allineamento spaziale	Robusto	Più sensibile
Accuratezza	Molto alta (99.1%)	Molto alta (99.56%)

3.2.7 Student-Teacher Avanzati (2022-2023)

Il paradigma student-teacher, introdotto inizialmente con metodi come Uninformed Students (2020), ha subito un'evoluzione significativa tra il 2022 e il 2023, producendo metodi più performanti e efficienti.

Reverse Distillation (2022). Reverse Distillation (RD) [102] modifica il paradigma student-teacher invertendo il flusso dell'informazione. Invece di teacher e student entrambi encoder con architetture simili, in RD il **teacher** è un *encoder* pre-addestrato e congelato (down-sampling), mentre lo **student** è un *decoder* (up-sampling) che deve ricostruire le feature estratte dal teacher partendo da una rappresentazione compressa (Figura 3.15).

Architettura e Inversione del Paradigma. Oltre all'inversione dei ruoli, si ha anche l'inversione del flusso dei dati: l'immagine va al teacher mentre lo student riceve solo un embedding compatto. La distillazione avviene trasferendo la conoscenza da rappresentazioni high-level (astratte) a low-level (dettagli), invertendo l'ordine naturale.

1. **Teacher Encoder** E_T (pre-trained, fixed): Una rete pre-addestrata su ImageNet (es. ResNet) che estrae feature multi-scala $\{F_1^T, F_2^T, F_3^T\}$ da un'immagine di input x.

2. **One-Class Bottleneck Embedding (OCBE)**: Modulo addestrabile che riceve le feature multi-scala dal teacher, le fonde tramite Multi-scale Feature Fusion (MFF) e le comprime in un **embedding compatto** $z \in \mathbb{R}^d$ con $d \ll C \times H \times W$. Questo crea un *collo di bottiglia informativo* (d piccolo) che forza la rete a estrarre solo l'informazione essenziale dei dati normali.

3. **Student Decoder** D_S (trainable): Una rete simmetrica ma inversa rispetto al teacher, che riceve l'embedding z e tenta di *ricostruire le feature originali* del teacher a ciascun livello: $\{\hat{F}_1^S, \hat{F}_2^S, \hat{F}_3^S\} = D_S(z)$.

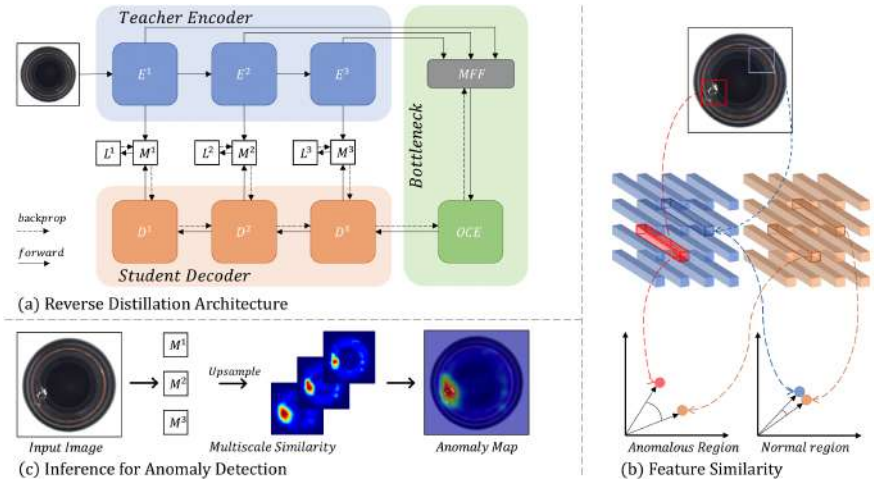

Figure 3.15: Reverse Distillation framework per anomaly detection. (a) Architettura: teacher encoder E pre-addestrato, One-Class Bottleneck Embedding (OCBE), student decoder D (addestrabile). Multi-scale Feature Fusion (MFF) combina le feature di basso e alto livello provenienti da E e le mappa in un codice compatto tramite il blocco One-Class Embedding (OCE). Durante il training, lo student impara a replicare le feature del teacher minimizzando la similarity loss $\mathcal{L}$. (b) Inferenza: il teacher estrae feature fedelmente, mentre lo student produce versioni senza anomalie. Una bassa similarità tra feature corrispondenti indica anomalia. (c) La predizione finale aggrega mappe di similarità multi-scala M. Fonte: [102]

Il bottleneck è cruciale: formula le anomalie come "perturbazioni sui pattern normali", e l'embedding compatto *filtra* queste perturbazioni, impedendo loro di propagarsi allo student.

Training e Loss. Addestrato solo su immagini normali, il sistema minimizza la dissimilarità tra feature del teacher e ricostruzioni dello student. La loss totale è la somma su tre livelli di una **cosine similarity loss negativa** (rende il modello più robusto alle variazioni delle feature, concentrandosi sulla *direzione*):

$$\mathcal{L}_{RD} = \sum_{k=1}^{3} \left[1 - \frac{1}{H_k W_k} \sum_{i,j} \text{cosine-sim} \left(F_k^T(i,j), \hat{F}_k^S(i,j) \right) \right] \tag{3.13}$$

dove cosine-sim$(a,b) = \frac{a^T b}{\|a\|\|b\|}$.

Rilevamento anomalie.

- **Per immagini normali**: Le feature F_k^T rientrano nella distribuzione appresa. L'OCBE le comprime bene, il decoder le ricostruisce fedelmente, determinando una bassa loss $\mathcal{L}_{RD}$
- **Per immagini anomale**: Le feature F_k^T sono "fuori distribuzione". L'embedding ϕ risulta distorto, portando a una ricostruzione scadente e ad una alta loss $\mathcal{L}_{RD}$

L'anomaly score pixel-level è la mappa di discrepanza:

$$A(\mathbf{x}) = \sum_{k=1}^{3} \text{Upsample} \left(1 - \frac{\mathbf{F}_k^T \cdot \mathbf{F}_k^S}{\|\mathbf{F}_k^T\| \|\mathbf{F}_k^S\|} \right) \tag{3.14}$$

Innovazioni e Vantaggi.

- **Non-similarity**: mentre il teacher (encoder) agisce da *filtro down-sampling*, estraendo feature sempre più astratte, lo student (decoder) ricostruisce da astratto a concreto (*filtro up-sampling*). Questa eterogeneità evita il problema dei "non-distinguishing filters" dove teacher e student, se troppo simili, tendono a convergere anche su anomalie
- **Bottleneck come filtro**: La bassa dimensionalità d dell'embedding agisce da *filtro informativo*. Informazioni anomale (rumore, anomalie) tendono a essere perse nella compressione.
- **Efficienza**: Il teacher è congelato. Si addestrano solo l'OCBE (pochi parametri) e il decoder.

Limitazioni. La performance di RD dipende fortemente dalla scelta della dimensione del bottleneck d e dalla capacità del teacher pre-addestrato di estrarre feature discriminative.

Reverse Distillation Revisited (RD++, 2023). Tien et al. [103] identificano alcune limitazioni critiche nell'originale Reverse Distillation (RD) e propongono RD++, un'architettura migliorata che le affronta attraverso meccanismi espliciti di supervisione (Figura 3.16).

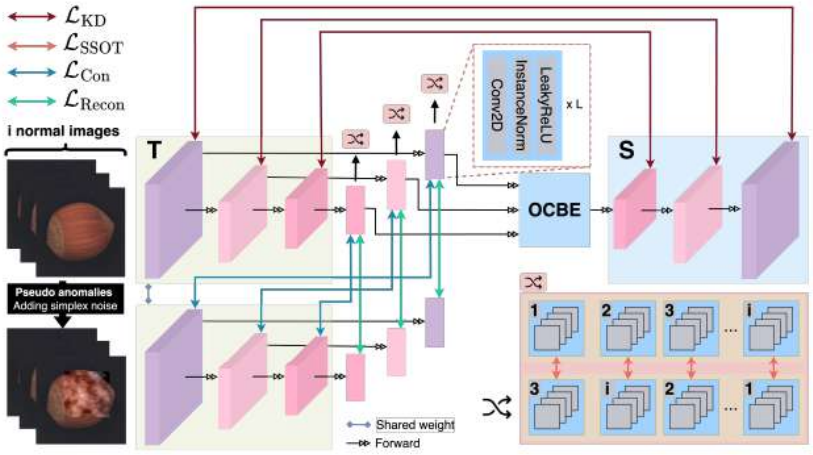

Figure 3.16: Overview di RD++ durante training. Rispetto a RD, RD++ integra dei livelli di proiezione (*projection layers*) dopo ogni blocco teacher intermedio per fornire rappresentazioni compatte prive di anomalie. La distillation loss $\mathcal{L}_{KD}$ viene combinata con: (1) Self-supervised Optimal Transport loss $\mathcal{L}_{SSOT}$, per proiettare lo spazio delle feature normali in una rappresentazione compatta, (2) Contrast loss $\mathcal{L}_{Con}$, per separare feature normali proiettate da quelle anomale, (3) Reconstruction loss $\mathcal{L}_{Recon}$ con pseudo-anomalie per guidare i projection layers a ricostruire lo spazio normale da feature pseudo-anomale. Fonte: [103]

Limitazioni di RD Originale. Analizzando l'output del modulo OCBE (input allo student), gli autori scoprono due problemi critici:

Feature Compactness Insufficiente: L'embedding prodotto per immagini normali presenta un'elevata varianza, indicando che il bottleneck non concentra efficacemente le feature nella regione normale dello spazio latente

Nessun Meccanismo Esplicito per Soppressione Anomalie: RD assume implicitamente che il bottleneck filtri le anomalie, ma manca un meccanismo di supervisione diretta che penalizzi la propagazione di feature anomale.

Architettura e Training Migliorati. RD++ introduce:

1. **Projection Layers Multi-Scala**: Strati convoluzionali leggeri inseriti dopo ogni blocco intermedio del teacher encoder, prima del modulo OCBE. Questi layers proiettano le feature in uno spazio più adatto prima della compressione nel bottleneck, permettendo di processare informazioni a diverse scale spaziali.

2. **Self-Supervised Optimal Transport Loss ($\mathcal{L}_{SSOT}$)**: Una nuova loss che forza le feature di diversi campioni normali ad essere vicine nello spazio latente, migliorando esplicitamente la compattezza della rappresentazione.

3. **Reconstruction Loss ($\mathcal{L}_{Recon}$)**: Nell'addestramento, vengono generate immagini sintetiche corrotte con rumore strutturato (Simplex noise). La loss incoraggia projection layers a mappare feature normali e pseudo-anomale *vicine*, insegnando ai projection layer a "ricostruire" feature normali anche da input alterati, sviluppando robustezza contro anomalie.

4. **Contrast Loss ($\mathcal{L}_{Con}$)**: Loss margin-based che crea una separazione esplicita nello spazio latente tra feature normali (proiettate) e feature anomale (non proiettate). Impedisce al modello di collassare tutte le feature in un'unica regione, mantenendo una distinzione tra pattern normali e anomali.

La loss totale combina questi termini con la distillation loss originale:

$$\mathcal{L}_{totale} = \mathcal{L}_{KD} + \alpha\mathcal{L}_{SSOT} + \beta\mathcal{L}_{Recon} + \gamma\mathcal{L}_{Con} \tag{3.15}$$

Risultati e Analisi. RD++ ottiene miglioramenti significativi rispetto a RD:

- **MVTec AD**: 99.44% image AUROC (+0.98% vs RD), 98.25% pixel AUROC.
- **Compattezza migliorata**: La varianza intra-classe delle feature normali si riduce.
- **Soppressione migliorata**: La separazione tra feature normali e anomale migliora.

L'ablation study dimostra che il guadagno principale (circa +0.87%) deriva dalla loss di compattezza (SSOT), mentre il contributo del training con pseudo-anomalie è più marginale ma complementare.

Contributo Principale. RD++ dimostra che la supervisione esplicita per la compattezza delle feature normali è cruciale per performance elevate nell'anomaly detection. Il metodo mantiene l'efficienza computazionale di RD e stabilì un nuovo state-of-the-art nel 2023.

EfficientAD (2024). EfficientAD [104] rappresenta l'ultima evoluzione del paradigma student-teacher per l'anomaly detection industriale. Il suo contributo principale è dimostrare che un *ensemble di approcci complementari semplici* può superare in accuratezza ed efficienza architetture singole più complesse, diventando il nuovo state-of-the-art sia nei benchmark che in scenari reali.

Architettura Tripartita. EfficientAD combina **tre componenti indipendenti** con ruoli complementari:

1. **Student-Teacher con PDN Teacher**: inizialmente, un **Patch Description Network (PDN)** (Figura 3.17) leggero (tra 1M e 2M di parametri) viene "distillato" da un encoder pre-addestrato grande (Wide-ResNet-101 su ImageNet). Questo processo di distillazione comprime la conoscenza della rete grande nel PDN piccolo. Poi, il PDN distillato diventa il **teacher** fisso, e uno **student** ancora più piccolo viene addestrato a replicarne l'output solo su dati normali. Al test, un'alta discrepanza tra student e PDN teacher indica una potenziale anomalia. Questo componente è particolarmente efficace nel rilevare *anomalie strutturali e semantiche*

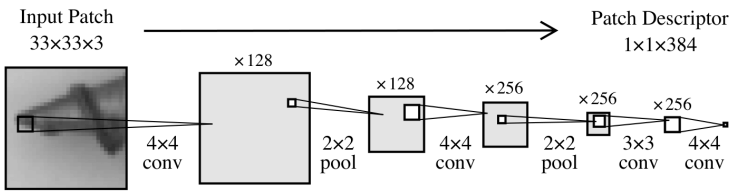

Figure 3.17: Architettura del Patch Description Network (PDN) in EfficientAD. Il PDN riceve patch 33×33×3 e produce patch descriptors 1×1×384 attraverso layer convoluzionali (4×4 conv, 4×4 conv, 4×4 conv, 3×3 conv, 4×4 conv) alternati a pooling 2×2. Questo teacher leggero fornisce feature dense per il training dello student ancora più compatto, permettendo anomaly detection real-time efficiente. Fonte: [104]

2. **Autoencoder Compatto**: Un autoencoder con bottleneck molto stretto (64 canali) è addestrato a ricostruire immagini normali. La forte compressione impedisce la ricostruzione fedele di patterm mai visti durante il training (anomalie): la ricostruzione fallisce su anomalie ma riesce su normali. Al test, un elevato errore di ricostruzione segnala un'anomalia. Questo componente è specializzato nel catturare *anomalie di texture e apparenza locale.*

3. **Normalizzazione delle Mappe di Anomalia**: Meccanismo di post-processing che combina e calibra gli output dei primi due componenti. Prende le mappe di anomalia grezze prodotte dallo Student-Teacher (A_{ST}) e dall'Autoencoder (A_{AE}), le normalizza statisticamente e le combina per produrre una mappa finale più robusta e meno sensibile a falsi positivi. Il processo include un passo di penalizzazione che riduce gli score in regioni

dove i due componenti sono in disaccordo, migliorando la calibrazione dello score finale.

Ensemble Finale. La mappa di anomalia finale viene ottenuta attraverso una combinazione pesata delle mappe normalizzate prodotte dai due componenti principali. I pesi di questa combinazione vengono ottimizzati empiricamente su un set di validazione.

Ottimizzazioni Efficienza.

- **Elaborazione Parallela**: I tre componenti (ST, AE, normalizzazione) sono indipendenti e possono essere eseguiti in parallelo sulla GPU.
- **Elaborazione a Bassa Risoluzione**: Le immagini vengono elaborate a risoluzioni ridotte (es. 256x256) senza penalizzare le performance, grazie alla progettazione efficiente delle reti.

Perché EfficientAD è Efficace.

- **Complementarità dei Segnali**: Lo Student-Teacher è sensibile a cambiamenti semantici e strutturali, mentre l'Autoencoder è sensibile a variazioni di texture e aspetto locale. La loro combinazione copre un ampio spettro di possibili difetti.
- **Efficienza Radicale**: L'attenzione alla leggerezza dei modelli (pochissimi parametri, quantizzazione) lo rende adatto al deployment in tempo reale anche su hardware limitato.
- **Semplicità Concettuale**: Non richiede la generazione di dati sintetici né complessi meccanismi di training addizionali, semplificando l'implementazione e la riproducibilità.

EfficientAD rappresenta un punto di arrivo significativo, dimostrando che l'intelligenza del sistema risiede più nell'architettura dell'ensemble e nell'ottimizzazione dell'efficienza che nella complessità di un singolo modello.

3.2.8 Metodi Semplificati (2023)

Mentre metodi come EfficientAD hanno dimostrato l'efficacia degli ensemble complessi, una tendenza alternativa e contemporanea ha esplorato la direzione opposta: la **semplicità radicale**. L'idea fondamentale è che la complessità architettuale, oltre un certo punto, non porta benefici misurabili in accuratezza, ma introduce overhead computazionale, fragilità nella scelta degli iperparametri e difficoltà di debugging e manutenzione [105].

SimpleNet (2023). SimpleNet [105] riduce al minimo la complessità, proponendo un sistema robusto, riproducibile ed efficiente.

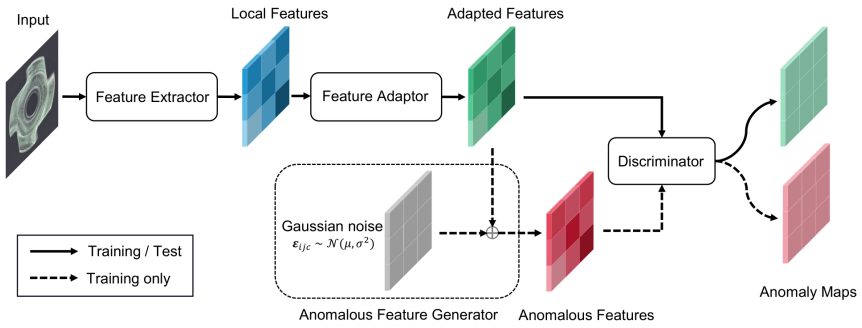

Figure 3.18: Architettura di SimpleNet. Nella fase di addestramento, campioni normali passano attraverso un Feature Extractor pre-addestrato per ottenere le feature locali (*local features*). In seguito, un Feature Adaptor adatta le feature al dominio target. Un Anomalous Feature Generator sintetizza feature anomale aggiungendo rumore Gaussiano $\varepsilon_{ijc} \sim \mathcal{N}(\mu, \sigma^2)$ alle features adattate. Il Discriminator impara a distinguere feature adattate (positive) da feature anomale (negative). All'inferenza, il generatore viene rimosso e il discriminatore produce direttamente anomaly maps. Fonte: [105]

Architettura: Four Components Only. SimpleNet si compone di solo 4 componenti essenziali (Figura 3.18):

1. **Feature Extractor (pre-trained, frozen)**: Una rete pre-addestrato su ImageNet (tipicamente WideResNet50). Pesi completamente frozen, senza fine-tuning, nessun backpropagation attraverso l'encoder. Estrazione multiscala da layer intermedi.

2. **Feature Adaptor (opzionale, leggero)**: Una o due convoluzioni 1×1 per ridurre la dimensionalità e allineare le feature alla distribuzione del task specifico. Questo è il modulo che "adatta" le feature pre-trained al dominio target.

3. **Anomalous Feature Generator**: SimpleNet genera anomalie *direttamente nello spazio delle feature*. Durante il training, a ciascun vettore di feature normale $\mathbf{q}_{h,w} \in \mathbb{R}^C$ viene aggiunto rumore Gaussiano $\epsilon \sim \mathcal{N}(\mu, \sigma^2)$:
$$\mathbf{q}^-_{h,w} = \mathbf{q}_{h,w} + \epsilon \tag{3.16}$$
Questa perturbazione sfrutta la compattezza dello spazio delle feature normali, spingendo le feature modificate al di fuori di questa regione per creare esempi anomali efficaci.

4. **Discriminator**: Un classificatore a due livelli (2-layer MLP) che agisce come *normality scorer*, stimando la probabilità che una feature sia normale. Per ogni posizione spaziale (h, w), riceve in input $\mathbf{q}_{h,w}$ (feature normale) o $\mathbf{q}^-_{h,w}$ (feature anomala) e produce uno score $D_\psi(\mathbf{q}_{h,w}) \in \mathbb{R}$.

Durante il training, impara a massimizzare lo score per feature normali e minimizzarlo per feature anomale.

Perché SimpleNet Funziona. Quattro insights fondamentali rivelati:

1. **Pre-trained features**: WideResNet50 addestrato su ImageNet apprende rappresentazioni semantiche e strutturali potenti. Il task di anomaly detection diventa "solo" imparare un boundary lineare nello spazio feature.
2. **Synthetic feature-space anomalies** : le anomalie generate nello spazio compatto delle features forniscono *contrasto sufficiente* con le features normali per supervised learning efficace.
3. **Costi di complessità**: Oltre un certo punto la complessità architetturale aggiunge overhead computazionale, fragilità sugli hyperparameters e difficoltà di debugging, senza benefici misurabili in accuratezza.

Limitazioni. SimpleNet presenta comunque alcune limitazioni da considerare. Per iniziare, la sua performance dipende fortemente dalla qualità del backbone pre-addestrato utilizzato come estrattore di feature. Inoltre, la strategia di generazione delle anomalie basata sulla semplice perturbazione con rumore Gaussiano nello spazio delle feature, sebbene efficace, potrebbe non catturare la piena varietà e complessità dei difetti reali incontrati in scenari industriali specifici. Benchè competitivo, su alcuni dataset specializzati metodi più complessi come EfficientAD o PatchCore riescono a ottenere un margine di accuratezza superiore, attestandosi intorno all'1-2% in più nelle metriche di punta. Infine, la mancanza di meccanismi di adattamento rende l'architettura minimalista meno flessibile nell'incorporare conoscenze specifiche del dominio (domain-specific knowledge) rispetto ad approcci più complessi e personalizzabili.

3.2.9 Vision-Language Models (2023-2024)

L'integrazione di vision-language models (VLM) pre-addestrati nell'anomaly detection rappresenta un cambio di paradigma in grado di risolvere limitazioni critiche dei metodi puramente visivi [106], [107]. Questi metodi sfruttano la conoscenza semantica multimodale, appresa dall'analisi congiunta di coppie immagine-testo, per superare le carenze dei metodi puramente visivi, soprattutto in termini di generalizzazione e adattamento a nuovi domini con pochi esempi.

Motivazioni e Limitazioni dei Metodi Vision-Only. I metodi tradizionali, estremamente accurati in condizioni ideali, possono presentare alcune vunerabilità critiche in scenari pratici:

1. **Domain specificity**: Feature estratte da reti pre-addestrate su ImageNet catturano bene pattern visivi generici (texture, edges, forme) ma mancano di una comprensione semantica profonda [1].

2. **Few-shot fragility**: La costruzione di rappresentazioni robuste della normalità richiede tipicamente un numero molto elevato di esempi. Con pochi esempi le performance degradano [81].

3. **Zero-shot impossibility**: In assenza di esempi visivi dal dominio target, i metodi basati solo sulla visione non hanno alcun meccanismo per trasferire la nozione di "anomalia". Senza supervisione visiva diretta, il loro comportamento equivale a un'ipotesi casuale.

Cosa Portano i Vision-Language Models. L'introduzione di modelli multimodali pre-addestrati come CLIP [108] affronta queste limitazioni sfruttando una conoscenza semantica fondata sia sulla visione che sul linguaggio, appresa da dataset web-scale

- **Radicamento semantico (semantic grounding)**: Gli spazi di embedding allineati permettono di specificare *concettualmente* cosa cercare ("defect", "anomaly", "damage", "scratch") andando oltre il confronto statistico di pattern visivi. Questo consente il trasferimento di conoscenza concettuale tra domini anche molto diversi [107].

- **Transfer via language**: Conoscenza di "anomalia" appresa su un dominio (es. "crepa su bottiglia di vetro") può trasferire ad altro dominio (es. "graffio su lamiera metallica") via descrizioni testuali condivise ("linear defect"), anche senza alcun esempio visivo del secondo dominio [106].

CLIP Foundation. CLIP (Contrastive Language-Image Pre-training) [108] addestra congiuntamente un image encoder (Vision Transformer o ResNet) e un text encoder (Transformer) a massimizzare la *cosine similarity* per coppie (immagine, testo) corrette e minimizzarla per incorrette. Dopo un pre-training su circa 400 milioni di coppie, CLIP fornisce **embedding allineati** in uno spazio condiviso: immagini e descrizioni testuali semanticamente simili risultano vicine.

Questa proprietà abilita la **classificazione zero-shot** tramite il semplice confronto con prompt testuali (ad esempio, confrontando la similarità con *"a photo of a defect"* e *"a photo of a normal object"*), senza alcun training specifico sul dominio target.

WinCLIP (2023). WinCLIP [106] è stato il primo metodo ad adattare CLIP all'anomaly detection attraverso un approccio a *sliding windows* per localizzazione pixel-level (Figura 3.19).

Data un'immagine, WinCLIP estrae le feature per ogni finestra e calcola gli embedding CLIP. Questo processo genera una mappa densa di feature, dove ogni posizione è rappresentata da un vettore che codifica l'informazione visiva locale, ben allineata con lo spazio semantico del linguaggio.

Per ottenere lo score di anomalia finale, il metodo combina i risultati delle singole finestre. Lo score di ciascuna finestra (dissimilarità tra la sua feature e l'embedding testuale del prompt che descrive la "normalità") viene prima

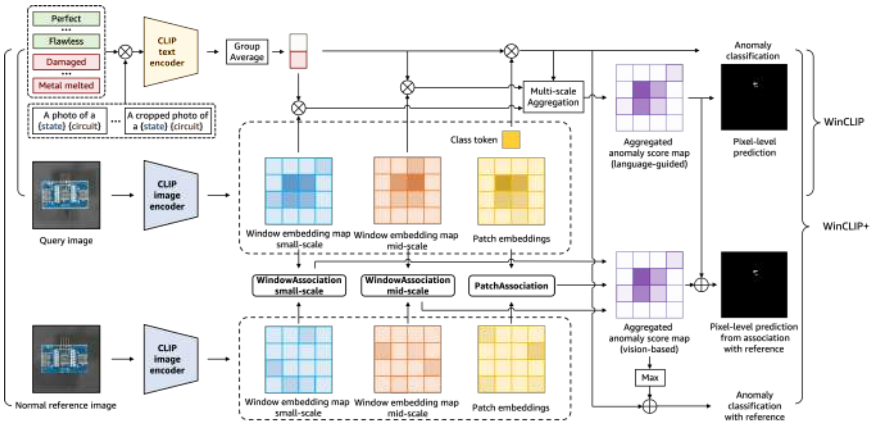

Figure 3.19: Workflow di WinCLIP e WinCLIP+. Templates testuali ("Perfect", "Flawless", stati anomali) vengono convertiti in due embedding testuali come prototipi di classe tramite l'encoder testuale di CLIP. WinCLIP (pannello superiore): feature multi-scala dell'encoder immagine di CLIP vengono correlate con i prototipi per zero-shot anomaly classification/segmentation. WinCLIP+ (pannello completo): aggiunge l'associazione di riferimento a livello di patch e finestre piccole/medie (Patch/WindowAssociation) per ottenere mappe di anomalia vision-based, che vengono poi aggregate per la segmentazione e il rilevamento di anomalie few-shot insieme ai punteggi guidati dal linguaggio. Fonte: [106]

riportato a livello di pixel. Successivamente, i contributi di tutte le finestre che sovrappongono uno stesso pixel vengono aggregati tramite una *media armonica*, privilegiando i segnali meno anomali. Per rilevare difetti di dimensioni diverse, WinCLIP utilizza infine finestre di dimensioni multiple (piccola, media e a scala d'immagine), fondendo le relative mappe di anomalia in un'unica predizione finale robusta e multi-scala.

- **Vantaggi**: Architettura zero-shot che non richiede training, interpretabilità data dall'uso di prompt testuali, flessibilità nell'adattarsi a nuove categorie di oggetti
- **Limitazioni**: Performance sensibili alla formulazione dei prompt testuali, un divario di accuratezza rispetto ai metodi supervisionati addestrati su dati specifici, e un costo computazionale non trascurabile dovuto all'elaborazione di molte finestre scorrevoli.

AnomalyCLIP (2024). AnomalyCLIP [107] nasce per adattare CLIP al **zero-shot anomaly detection** (ZSAD): rilevare anomalie *senza alcun esempio dal dominio target*, introducendo una serie di innovazioni che portano zero-shot a livelli competitivi (Figura 3.20).

- **Object-Agnostic Prompts**: Abbandona i prompts legati a specifiche

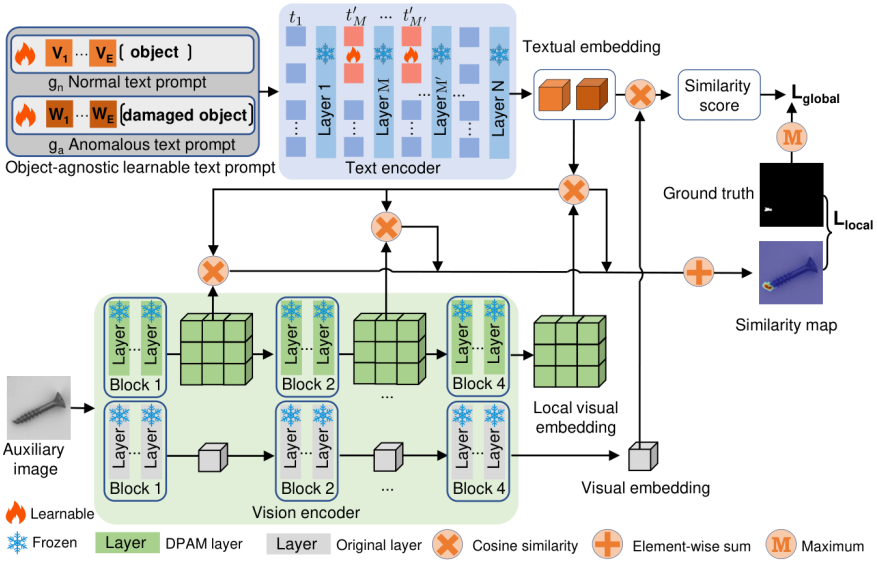

Figure 3.20: Overview di AnomalyCLIP. Per adattare CLIP a zero-shot anomaly detection (ZSAD), AnomalyCLIP introduce template di prompt testuali agnostici ($\mathbf{v}_1 \cdots \mathbf{v}_E$ per normale, $\mathbf{w}_1 \cdots \mathbf{w}_E$ per anomalo) che catturano normalità e anomalia generiche indipendentemente dalla semantica dell'oggetto. La glocal context optimization incorpora la semantica di anomalia sia a livello globale che di dettaglio nell'apprendimento dei prompt testuali agnostici rispetto all'oggetto. Textual prompt tuning e DPAM (Directed Patch Attention Module, verde) in layer selezionati del vision encoder abilitano l'apprendimento negli spazi visivi testuali e locali. Fonte: [107]

categorie in favore di prompts generici ("image without defects") , eliminando necessità di conoscere la classe dell'oggetto.

- **Global-Local Dual Alignment**: Introduce una supervisione a due livelli: oltre ad allineare l'embedding dell'intera immagine con descrizioni testuali, allinea anche le feature a livello di pixel con etichette di normalità/anomalia, garantendo così una comprensione sia del contesto globale che del dettaglio locale per la segmentazione.

- **Multi-Layer Local Optimization**: Invece di utilizzare feature da un singolo layer, AnomalyCLIP applica la local context optimization a *multiple intermediate layers* dell'encoder visivo di CLIP. Per ogni layer selezionato M_l, calcola mappe di similarità locali S_{n,M_l} e S_{a,M_l} tra i prompt testuali e le feature visive a quel livello. Queste mappe da layer a diversa profondità (che catturano informazioni a diversa granularità spaziale/semantica) vengono poi combinate per formare la predizione finale, permettendo di rilevare sia difetti fini che anomalie strutturali su larga scala

Vantaggi e Limitazioni.

- **Generalizzazione Superiore**: La conoscenza semantica multimodale trascende le semplici statistiche visive, consentendo un trasferimento più robusto tra domini diversi [106], [107].

- **Few-Shot Efficacia**: Questi modelli risultano particolarmente adatti a scenari reali, dove la raccolta di grandi dataset annotati può essere difficile.

- **Limitazioni**: Overhead computazionale e un tetto di accuratezza legegrmente inferiore ne limitano l'adozione in applicazioni critiche per velocità o precisione. La loro performance è inoltre sensibile all'ingegneria dei prompt.

3.2.10 Confronto

Dopo aver esaminato l'evoluzione dei metodi, presentiamo un confronto sintetico delle caratteristiche chiave e delle performance relative. I valori di riferimento sono basati sul benchmark **MVTec AD** e derivano dai paper originali quando disponibili. Si noti che le performance assolute possono variare leggermente tra diverse implementazioni e configurazioni hardware.

Panorama Sintetico dei Metodi

La Tabella 3.2 riporta un confronto sintetico tra i metodi più rilevanti discussi in questo capitolo, evidenziando le loro caratteristiche principali e performance indicative sul benchmark MVTec AD.

Table 3.2: Confronto sintetico dei principali metodi (valori indicativi su MVTec AD)

Metodo	Anno	I-AUROC	P-AUROC	Tipo	Tr.	Vel.
PaDiM	2020	97.9	98.3	Feature	No	Media
SPADE	2021	85.5	96.0	k-NN	No	Lenta
CutPaste	2021	95.2	96.8	Synthetic	Sì	Veloce
DRAEM	2021	98.4	98.6	Rec+Disc	Sì	Media
FastFlow	2021	99.4	98.5	Flows	Sì	Media
PatchCore	2022	99.1	98.1	Memory	No*	Media
MemSeg	2022	99.6	98.8	Segm.	Sì	**Veloce†**
RD++	2023	99.4	98.3	S-T	Sì	Media
SimpleNet	2023	99.2	98.1	Discrim.	Sì	**Veloce†**
EfficientAD	2024	99.5	98.7	Ensemble	Sì	Veloce
WinCLIP	2023	91.8	95.2	V-L	**No**	Lenta
AnomalyCLIP‡	2024	95-98	93-96	V-L	Opz.	Media

Note: Valori in %. *PatchCore richiede costruzione coreset offline. †Molto veloce (<10ms). ‡AnomalyCLIP: zero-shot (85-88), few-shot (95-96), full-shot (97-98). Velocità relativa su GPU moderna: Veloce† (<10ms), Veloce (10-50ms), Media (50-150ms), Lenta (>150ms). Abbreviazioni: I-AUROC (Image AUROC), P-AUROC (Pixel AUROC), Tr. (Training), S-T (Student-Teacher), V-L (Vision-Language), Rec+Disc (Ricostruzione+Discriminatore), Segm. (Segmentazione).

Guida alla Selezione per Scenario Applicativo

La scelta del metodo ottimale dipende fortemente dal contesto applicativo. La Tabella 3.3 sintetizza i punti di forza di ciascun approccio.

Tendenze Evolutive e Considerazioni Finali

L'analisi comparativa rivela cinque tendenze chiare che hanno plasmato il campo tra il 2017 e il 2024:

1. **Semplicità efficace**: I metodi moderni (SimpleNet, EfficientAD) dimostrano che architetture minimaliste ma ben progettate possono superare sistemi complessi (GAN, autoencoder profondi).

2. **Ensemble complementari**: EfficientAD ha stabilito che combinare componenti semplici con punti di forza complementari (struttura, texture, robustezza) supera le architetture monolitiche.

3. **Memoria esplicita**: I memory-based methods (PatchCore, MemSeg) hanno mostrato che memorizzare esplicitamente pattern normali è più efficace dell'apprendimento implicito.

4. **Conoscenza multimodale**: I Vision-Language Models (WinCLIP, AnomalyCLIP) affrontano limitazioni fondamentali dei metodi puramente visivi, specialmente in scenari few-shot e multi-dominio.

5. **Efficienza**: La pressione verso il deployment industriale ha reso l'efficienza computazionale (velocità, memoria) un criterio di progetto primario.

Table 3.3: Guida pratica alla selezione del metodo

Scenario Prioritario	Metodo Consigliato e Motivazione
Accuratezza Massima	**EfficientAD** (ensemble ottimizzato) o **RD++** (student-teacher avanzato) per performance top-tier (>99.4% AUROC). **PatchCore** se non si può addestrare.
Inferenza Real-Time	**SimpleNet** (estremamente leggero) o **MemSeg** (segmentazione diretta) per latenze <10ms. **EfficientAD** per miglior bilanciamento accuratezza/velocità.
Dati Limitati (Few-Shot)	**AnomalyCLIP** (eccelle con K<10 esempi) sfruttando conoscenza semantica pre-esistente. **RD++** per un approccio solamente visivo.
Validazione Concettuale Rapida	**SimpleNet** (implementazione minimalista) o **PatchCore** (nessun training). Consentono validazione concettuale in tempi brevi.
Generalizzazione Cross-Domain	**AnomalyCLIP** (migliore robustezza a cambiamenti di dominio) grazie all'allineamento visione-linguaggio.
Localizzazione Precisa	**MemSeg** (segmentazione end-to-end) o **DRAEM** (ricostruzione discriminativa) per mappe di anomalia dettagliate.
Zero-Shot / Nessun Training	**WinCLIP** (pronto all'uso) o **PatchCore** (solo costruzione coreset).

Conclusioni. Non esiste un metodo universalmente ottimale per l'anomaly detection industriale. La scelta deve bilanciare accuratezza, efficienza computazionale, quantità di dati disponibili e requisiti di deployment. La tendenza attuale privilegia architetture semplici, efficienti e generalizzabili, con un ruolo crescente della conoscenza semantica multimodale per superare i limiti della supervisione puramente visiva. Per applicazioni critiche, si raccomanda sempre la validazione sperimentale diretta sul dominio specifico di interesse.

3.3 Metodi 3D per Anomaly Detection

L'ispezione visiva basata su immagini 2D, pur avendo raggiunto livelli di accuratezza notevoli con i metodi descritti nelle sezioni precedenti (si veda in particolare la Tabella 3.2), presenta limitazioni intrinseche nel rilevamento di difetti con caratteristiche tridimensionali. Deformazioni superficiali sottili, variazioni di profondità, ammaccature, graffi rilevabili solo da angolazioni multiple — tutti questi difetti non sono catturabili da una singola proiezione 2D. In molti contesti industriali, inoltre, l'ispezione visiva tradizionale viene affiancata o sostituita da sensori 3D (scanner a luce strutturata, laser, time-of-flight camera) in grado di acquisire la geometria completa dell'oggetto esaminato.

L'introduzione del benchmark MVTec 3D-AD [2] nel 2021 ha fornito il primo dataset standardizzato per l'anomaly detection su dati 3D, comprendente point cloud ad alta risoluzione e immagini RGB allineate per 10 categorie di oggetti. Questo dataset ha dato inizio allo sviluppo di una nuova generazione di metodi capaci di sfruttare informazioni geometriche, da soli o in combinazione con l'RGB.

I dati 3D introducono sfide specifiche rispetto all'analisi 2D:

- la natura non strutturata e sparsa dei point cloud richiede architetture dedicate (PointNet, voxel grid, sparse convolution) che possano operare su coordinate irregolari
- l'allineamento tra modalità RGB e 3D aggiunge complessità computazionale
- la densità di campionamento varia tra acquisizioni e all'interno dello stesso oggetto, rendendo meno diretta l'applicazione di metodi feature-based convenzionali come quelli descritti nel paragrafo 3.2.3

Nonostante queste difficoltà, i metodi 3D hanno mostrato progressi rapidi, beneficiando anche dell'integrazione con tecniche 2D consolidate estese. Nelle prossime sottosezioni verranno analizzati i principali approcci proposti in letteratura, distinguendo tra metodi che operano esclusivamente su dati 3D (point cloud, voxel) e metodi multimodali che combinano RGB e geometria.

3.3.1 Estensioni Dirette ai Dati 3D (2021)

I primi approcci all'anomaly detection su dati 3D adottarono la strategia più immediata: convertire i point cloud in rappresentazioni voxelizzate regolari e applicare adattamenti diretti dei metodi 2D già consolidati, mantenendo le stesse architetture con modifiche minime. Questi metodi furono proposti come baseline nel paper introduttivo di MVTec 3D-AD [2].

Voxel Autoencoder e Voxel f-AnoGAN. Il **Voxel Autoencoder** [109] estende il paradigma ricostruttivo degli AE 2D a volumi 3D voxelizzati: il modello viene addestrato a ricostruire voxel normali, utilizzando l'errore di ricostruzione come anomaly score. **Voxel f-AnoGAN** [110] applica analogamente l'architettura f-AnoGAN (si veda la Sezione 3.2.1) a rappresentazioni volumetriche.

I risultati su MVTec 3D-AD rivelarono performance modeste. Ad esempio, Voxel f-AnoGAN, il migliore tra i metodi solo-3D, ottenne un AUPRO di segmentazione di circa 0.583, mentre la sua versione con RGB integrato raggiunse lo 0.639 [2]. Questi risultati evidenziavano un divario significativo rispetto alle performance che i metodi 2D raggiungevano sui loro benchmark, confermando che la semplice voxelizzazione e l'applicazione di architetture 2D non erano sufficienti [2].

Le limitazioni sono quelle già osservate per i corrispettivi 2D (reconstruction bias, information loss) ma amplificate dalla discretizzazione voxelica, che introduce una perdita di risoluzione geometrica e una crescita cubica dei requisiti di memoria all'aumentare della risoluzione desiderata [111].

È interessante notare come entrambi i metodi provengano originariamente dal dominio dell'analisi di immagini mediche cerebrali (MRI) e siano stati adattati da Bergmann et al. al contesto industriale come baseline: questo riflette la

mancanza, nel 2021, di metodi nativamente progettati per point cloud industriali, e motiva direttamente lo sviluppo delle linee di ricerca descritte nelle sezioni successive.

Variation Models e Metodi su Mappe di Profondità. Come baseline aggiuntiva, il paper di MVTec 3D-AD valutò anche *variation models*, metodi applicati direttamente alle immagini di profondità (depth maps). In questo caso, si trattava il canale di profondità come una quarta banda di un'immagine RGB-D, utilizzando architetture 2D standard (come Autoencoder e GAN) [2]. I risultati furono persino inferiori a quelli dei metodi voxel: sebbene concettualmente semplice, questo approccio perde completamente l'informazione geometrica tridimensionale e si rivela inadeguato per anomalie che si manifestano esclusivamente nella struttura 3D dell'oggetto, come ammaccature profonde o deformazioni [2].

L'insieme di questi risultati deludenti confermò la necessità di metodi nativamente progettati per la geometria 3D, stimolando le linee di ricerca descritte nelle sezioni successive, che si concentrarono sull'estrazione di feature geometriche robuste.

3.3.2 Metodi Feature-based per Point Cloud (2022–2023)

Il trasferimento dell'intuizione dei metodi feature-based 2D (sfruttare rappresentazioni pre-addestrate potenti invece di addestrare generatori complessi) al dominio 3D ha prodotto i metodi più efficaci per l'analisi di point cloud. La differenza principale rispetto al caso 2D risiede nella necessità di backbone pre-addestrati specifici per point cloud, come PointMAE [112] o Point Transformer [113], in grado di estrarre feature geometriche significative.

BTF — Back to the Feature (2023). Alla pubblicazione, BTF (*Back to the Feature*) [114] divenne un importante riferimento per l'anomaly detection su point cloud. Il contributo centrale del paper è un'analisi empirica sistematica delle rappresentazioni 3D: confrontando descrittori hand-crafted (FPFH, HoG, SIFT su depth map) con metodi deep (PointNeXt, SpinNet), si mostra che la *rotation invariance* è la proprietà discriminante più importante.

Motivazione e Rotazione-Invarianza. I metodi che non garantiscono questa proprietà falliscono nel riconoscere anomalie geometriche quando l'oggetto si presenta in orientamenti diversi, mentre descrittori intrinsecamente invarianti alla rotazione (come FPFH) mantengono la loro efficacia indipendentemente dalla posa.

Feature Extraction. BTF opera direttamente sulla geometria del point cloud senza richiedere una voxelizzazione.

Per la componente 3D estrae descrittori **FPFH** (Fast Point Feature Histograms) [115]: per ogni punto, FPFH calcola prima i punti k-NN (*k-Nearest-*

Neighbors) rispetto al punto centrale della regione, per poi calcolare un istogramma in funzione delle normali di superficie e della distanza dai punti vicini. Per la componente colore, BTF applica PatchCore sull'immagine RGB corrispondente, estraendo feature da un WideResNet50 pre-addestrato su ImageNet. Le due rappresentazioni vengono concatenate, formando un descrittore combinato colore + 3D.

k-NN Scoring. Durante il training, BTF raccoglie i descrittori di tutti i punti di tutte le nuvole normali in un unico insieme $S = \{\phi(x, j)\ \forall x\ \forall j\}$, senza alcuna compressione o subsampling. Al test time, per ogni punto del campione di test si calcola la distanza dal suo k-esimo vicino più prossimo nell'insieme S: una distanza elevata indica che il punto si discosta dalla distribuzione normale appresa e segnala una potenziale anomalia. Questo produce direttamente una mappa di anomalia a livello di punto (*point-level segmentation*), mentre lo score a livello di campione è determinato dal punto con la distanza massima. La scelta di usare l'intero insieme S senza compressione è resa praticabile dalla bassa dimensionalità dei descrittori FPFH, molto più contenuta rispetto alle feature dense estratte da CNN.

Risultati e Impatto. BTF dimostrò che descrittori geometrici classici come FPFH, grazie alla loro invarianza alla rotazione, raggiungono performance competitive rispetto a metodi che sfruttano backbone profondi costosi, con il vantaggio di non richiedere alcuna fase di training. Il metodo stabilì un baseline solido per MVTec 3D-AD e rimane ampiamente utilizzato come confronto nei benchmark 3D più recenti (SiM3D [4], Real3D-AD [116]). La limitazione principale è la sensibilità alla densità e all'allineamento del point cloud: variazioni significative nel campionamento possono alterare le statistiche FPFH.

Shape-Guided Dual-Memory Learning (2023). Shape-Guided [117] introduce un framework a due esperti specializzati, ciascuno dedicato a una modalità, le cui memory bank sono collegate da un meccanismo di guida geometrica (Figura 3.21).

Shape Expert e SDF Memory Bank. Il point cloud viene suddiviso in 3D patch locali tramite Farthest Point Sampling (FPS). Per ciascun patch, PointNet [118] estrae un vettore di feature $\mathbf{f}$ che codifica la geometria locale. Questo vettore viene passato a una Neural Implicit Function (**NIF**) che apprende la *signed distance function* (**SDF**) della superficie: per ogni punto query $\mathbf{q}$ vicino alla superficie, il modello predice la sua distanza con segno

$$s = \varphi(\mathbf{q}; \mathbf{f}) \tag{3.17}$$

Un aspetto chiave è che la NIF φ è **condivisa e category-agnostic**: lo stesso modello viene addestrato su tutte le patch di tutte le categorie, senza specializzazione per singola classe. Di conseguenza, al termine del training è sufficiente memorizzare i soli vettori $\mathbf{f}$ di PointNet nella **SDF memory bank**

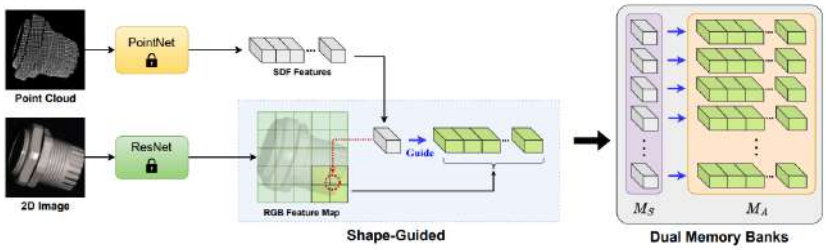

Figure 3.21: Dual memory bank di Shape-Guided. La SDF memory bank M_S memorizza le feature geometriche locali di ciascun patch normale. La RGB memory bank M_A non è un insieme piatto di descrittori, ma una collezione di dizionari RGB indicizzati per patch SDF: ogni entry di M_S punta al proprio dizionario di feature RGB corrispondenti in M_A. Questo collegamento shape-guided permette di cercare i nearest neighbor RGB solo nel sottoinsieme geometricamente coerente con la patch di test. Fonte: [117]

M_S, senza salvare i parametri della NIF per ciascuna patch. Il vantaggio rispetto a descrittori classici come FPFH è che le SDF modellano la geometria in modo continuo, catturando deformazioni sottili che istogrammi discreti non riescono a rappresentare.

Appearance Expert e RGB Memory Bank Shape-Guided. L'*appearance expert* costruisce la RGB memory bank M_A sfruttando il collegamento geometrico stabilito dallo shape expert. Per ogni patch SDF in M_S, i punti 3D corrispondenti sono proiettati sul piano immagine 2D per recuperare le feature RGB da un WideResNet50 pre-addestrato, espanse di due pixel per catturare anche i bordi dei difetti.

Ogni entry di M_S ha il proprio dizionario RGB dedicato in M_A: al test time, per un patch di test si trovano prima i vicini più prossimi in M_S (patch normali con geometria più simile). Solo per questi si accede ai rispettivi dizionari RGB in M_A: così, il confronto del colore avviene esclusivamente tra patch che condividono una struttura geometrica simile. Ad esempio, il colore di una zona curva dell'oggetto non viene mai confrontato con le feature RGB di una zona piatta, evitando falsi positivi.

Inferenza e Score Fusion. Per ogni patch di test si trovano i $k_1 = 10$ vicini più prossimi in M_S, che formano un dizionario locale. Invece di usare la distanza dal singolo nearest neighbor come anomaly score, si applica **sparse coding**: si cerca la combinazione lineare sparsa dei 10 elementi che meglio ricostruisce la feature di test, e la distanza di ricostruzione diventa lo score di anomalia geometrica. Lo stesso meccanismo viene poi applicato per la componente RGB usando il dizionario shape-guided estratto da M_A, producendo uno score di anomalia di colore.

Poiché le scale numeriche dei due score sono molto diverse, viene applicato

un allineamento score-map calibrato su 25 campioni normali: la media $\pm 3 \times$ standard deviation della distribuzione RGB viene mappata sulla corrispondente distribuzione SDF. Le due mappe di anomalia vengono infine fuse per massimo pixel-wise.

Risultati e Impatto. Shape-Guided superò BTF su MVTec 3D-AD (AU-PRO 0.976 vs 0.964), con il vantaggio aggiuntivo di richiedere meno memoria rispetto a BTF che usa PatchCore sull'intero insieme RGB: la struttura shape-guided riduce il numero di feature RGB da indicizzare da 208,230 a 26,452.

Reg3D-AD (2023). Reg3D-AD [116] è un metodo *registration-based* progettato come baseline per lo scenario few-shot del benchmark Real3D-AD, dove ogni categoria dispone di un numero limitato di prototipi per il training (4 o meno). La sfida è particolarmente difficile perché i prototipi di training sono scansioni complete a 360° dell'oggetto, mentre i campioni di test sono acquisiti da un solo lato, replicando la condizione reale di una linea di produzione in cui lo scanner è fisso e ispeziona una sola faccia del prodotto.

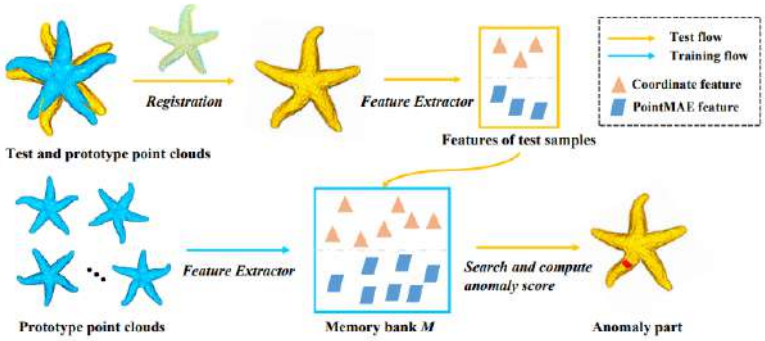

Figure 3.22: Pipeline di Reg3D-AD. In fase di training (frecce blu), le feature estratte dal training set vengono sottoposte a campionamento per selezionare quelle più rappresentative, che andranno a costituire la memory bank M. In fase di test (frecce gialle), il campione viene prima calibrato usando il prototipo come riferimento, poi le sue feature vengono estratte e confrontate con M per calcolare l'anomaly score punto per punto. Fonte: [116]

Registrazione e Feature Extraction. Per prima cosa, Reg3D-AD allinea il point cloud di test con il prototipo normale tramite RANSAC, riducendo le differenze dovute a variazioni di posa. Dopo l'allineamento, il metodo adotta una **dual-feature representation** ispirata a PatchCore [81]:

- **Feature locali**: le coordinate (x, y, z) di ciascun punto, che codificano la posizione e la struttura geometrica locale della superficie.

- **Feature globali**: feature estratte da PointMAE [112], che catturano la rappresentazione complessiva del prototipo nella sua interezza.

Entrambe le tipologie di feature vengono memorizzate in una feature memory bank costruita sui prototipi normali di training.

Risultati e Limitazioni. Reg3D-AD ottenne un object-level AUROC medio di 0.704 su Real3D-AD, superando significativamente tutti gli altri metodi del benchmark (BTF, M3DM, PatchCore nelle varie configurazioni), confermando quanto il few-shot e la disparità training/test tra scansione completa e parziale rendano il task difficile. La limitazione principale del metodo è la dipendenza dalla qualità dell'allineamento RANSAC: oggetti con geometria simmetrica o con superfici poco texturate possono generare allineamenti errati, propagando l'errore all'anomaly score.

3.3.3 Metodi Multimodali RGB+3D (2023)

Nonostante i buoni risultati ottenuti dai metodi precedenti, le informazioni di colore e texture contenute nelle immagini RGB restano complementari alla geometria 3D: ad esempio, un graffio superficiale può essere visibile nell'RGB ma impercettibile nella geometria, mentre un'ammaccatura può alterare la struttura 3D senza lasciare tracce evidenti nel colore. I metodi multimodali fondono entrambe le modalità per ottenere detection più robusta e completa [119].

AST — Asymmetric Student-Teacher (2022). AST (*Asymmetric Student-Teacher*) [120] risolve un problema dei metodi student-teacher precedenti: se student e teacher hanno architettura simile, lo student tende a generalizzare anche su campioni anomali, producendo un anomaly score basso (e non riconoscendo quindi l'anomalia). AST introduce un'asimmetria architettonica radicale per forzare la divergenza degli output sui campioni anomali.

Architettura Asimmetrica.

- Il **teacher** è un **normalizing flow** basato su Real-NVP addestrato a trasformare la distribuzione delle feature normali verso una distribuzione gaussiana $\mathcal{N}(0, I)$. La biettività del flow garantisce che, per input fuori distribuzione (anomalie), il teacher produca output divergenti rispetto ai campioni normali.
- Lo **student** è invece una rete feed-forward convenzionale, addestrata a replicare gli output del teacher sui soli campioni normali. La diversità architettonica impedisce allo student di imitare la divergenza del teacher sui campioni anomali, amplificando il segnale di anomalia.

Il teacher riceve le feature estratte da un backbone pre-addestrato su ImageNet. Lo student riceve le stesse feature a cui viene concatenata la mappa di profondità come canale aggiuntivo (tramite pixel-unshuffling per allinearne le

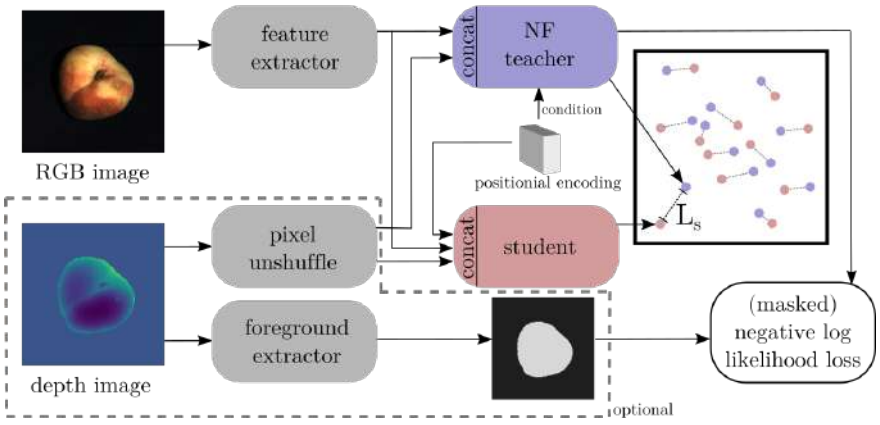

Figure 3.23: Pipeline di AST. Il teacher (normalizing flow, viola) e lo student (rete feed-forward, rosa) ricevono feature estratte da un backbone ImageNet; lo student riceve in aggiunta la mappa di profondità processata con pixel-unshuffle e una maschera di foreground estratta dai dati 3D. L'anomaly score L_s è la distanza tra gli output. Fonte: [120]

dimensioni). Il dato 3D viene usato anche per estrarre una maschera di foreground che esclude il background dal calcolo della loss (Figura 3.23).

Anomaly Score. Lo student viene addestrato a minimizzare la distanza tra i propri output e quelli del teacher sui campioni normali. Al test time, una distanza elevata segnala un'anomalia.

Il vantaggio rispetto all'uso diretto delle likelihood del flow come anomaly score è che eventuali stime di likelihood errate vengono compensate: se il flow assegna erroneamente alta likelihood a un campione anomalo, lo student non riuscirà comunque a replicare quell'output, mantenendo alto l'anomaly score.

Risultati. AST fu tra i primi metodi a sfruttare efficacemente la combinazione RGB+depth su MVTec 3D-AD, superando i baseline puramente 2D e dimostrando il valore della profondità come modalità complementare. Rimane un riferimento consolidato nei benchmark recenti [3].

M3DM — Multimodal 3D Anomaly Detection (2023). M3DM [121] rappresenta un approccio multimodale completo, combinando feature RGB e point cloud in un framework di memory bank unificato (Figura 3.24).

Feature Extraction e Point Feature Alignment. M3DM utilizza due backbone pre-addestrati separati:

- un **Point Transformer** [113] per le feature 3D

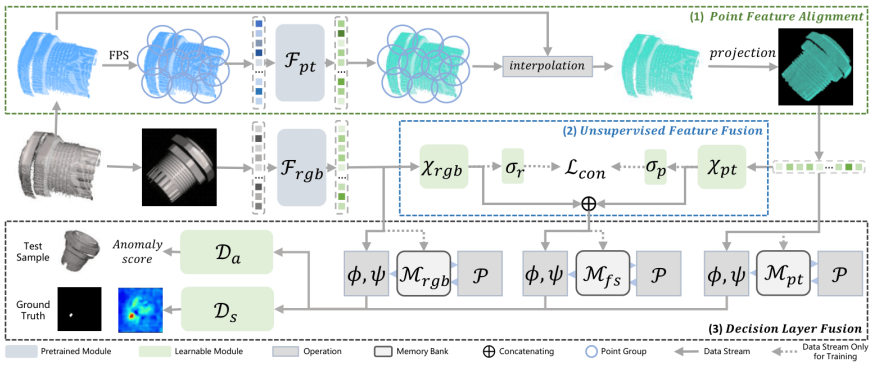

Figure 3.24: Pipeline di M3DM. (1) *Point Feature Alignment* (PFA): le feature del Point Transformer vengono allineate al piano immagine tramite interpolazione e proiezione. FPS indica il farthest point sampling e F_{pt} è un Point Transformer pre-addestrato. (2) *Unsupervised Feature Fusion* (UFF): le feature RGB e 3D vengono fuse tramite MLP con contrastive loss $\mathcal{L}_{con}$. $Frgb$ è un Vision Transformer, χ_{rgb} e χ_{pt} sono layer MLP, mentre σ_r e σ_p sono singoli layer fully connected. (3) *Decision Layer Fusion* (DLF): integra le informazioni multimodali attraverso multiple memory bank e produce la decisione finale mediante due moduli learnable $\mathcal{D}_a$ e $\mathcal{D}_s$ per detection e segmentation. $\mathcal{M}_{rgb}$, $\mathcal{M}_{fs}$, $\mathcal{M}_{pt}$ sono le memory bank, ϕ e ψ sono le funzioni di score per detection e segmentation con singola memory bank, e P rappresenta l'algoritmo di costruzione delle memory bank. Fonte: [121]

- un **Vision Transformer (ViT)** [122] per le feature RGB

Poiché il Point Transformer opera su gruppi di punti 3D, M3DM introduce una fase di **Point Feature Alignment** (PFA) che allinea le feature 3D del point cloud in una rappresentazione nello spazio 2D tramite interpolazione e proiezione, rendendole direttamente comparabili con le feature spaziali del ViT.

Unsupervised Feature Fusion e Tre Memory Bank. Invece di concatenare direttamente le feature delle due modalità (approccio che crea interferenze e performance peggiori), M3DM introduce una fase di **Unsupervised Feature Fusion** (UFF): due MLP proiettano le feature RGB e 3D in uno spazio comune, addestrati con una **patch-wise contrastive loss** $\mathcal{L}_{con}$ che incoraggia le feature della stessa posizione a essere vicine tra modalità diverse e quelle di posizioni diverse a essere lontane. Le feature fuse costituiscono una terza rappresentazione.

Le feature di tutte e tre le sorgenti (RGB, 3D, fusione) vengono memorizzate in **tre memory bank separate** $\mathcal{M}_{rgb}$, $\mathcal{M}_{pt}$, $\mathcal{M}_{fs}$ (costruite con coreset subsampling). Al momento del test, ogni memory bank definisce un anomaly score e una segmentation map indipendenti tra loro.

Decision Layer Fusion e Scoring. Al test time le distanze dai nearest neighbor vengono calcolate separatamente nelle tre memory bank, ottenendo tre

anomaly score indipendenti e analogamente tre segmentation map. La fusione finale è affidata a due moduli **learnable** $\mathcal{D}_a$ e $\mathcal{D}_s$, basati su OCSVM, che combinano i punteggi delle tre memory bank producendo rispettivamente l'anomaly score a livello immagine e la segmentation map.

Impatto e Limitazioni. M3DM stabilì un nuovo state-of-the-art su MVTec 3D-AD al momento della sua introduzione, confermando che la fusione di modalità complementari porta benefici misurabili rispetto ai metodi unimodali. Il metodo è ampiamente adottato come punto di confronto in benchmark successivi (Real-IAD D^3 [3], SiM3D [4]). Le limitazioni principali riguardano il costo computazionale (due backbone separati più la procedura di fusione) e la dipendenza dalla qualità dell'allineamento spaziale tra la nuvola di punti e le immagini RGB, che richiede una calibrazione accurata del setup di acquisizione.

CFM — Crossmodal Feature Mapping (2024). Mentre BTF e M3DM memorizzano grandi memory bank di feature multimodali, il che li rende accurati ma lenti e pesanti in memoria, e AST rinuncia alla struttura 3D riducendola ad un canale aggiuntivo, CFM (*Crossmodal Feature Mapping*) [123] propone un paradigma alternativo che elimina completamente le memory bank mantenendo performance state-of-the-art, sfruttando gli stessi backbone frozen di M3DM, DINO ViT-B/8 [124] per le feature RGB e Point-MAE [112] per le feature 3D.

Crossmodal Mapping come Segnale di Anomalia. In campioni normali esiste una **relazione** sistematica tra le feature RGB e le feature 3D della stessa posizione: una zona curva dell'oggetto ha sia una certa geometria nel point cloud sia una certa texture nell'immagine, e questa corrispondenza è apprendibile.

CFM impara questa relazione tramite due funzioni di mapping $\mathcal{M}_{2D\to3D}$ e $\mathcal{M}_{3D\to2D}$, implementate come MLP leggeri a tre layer, addestrati su campioni normali: $\mathcal{M}_{2D\to3D}$ apprende a predire le feature 3D a partire dalle feature RGB corrispondente, e $\mathcal{M}_{3D\to2D}$ l'opposto (Figura 3.25).

Un campione anomalo, per definizione, presenta combinazioni di feature RGB-3D mai osservate nel training. Al test time, il mapping non riesce a predire correttamente la feature dell'altra modalità: la discrepanza (distanza euclidea) tra la feature ricostruita $\hat{E}_{3D} = \mathcal{M}_{2D\to3D}(E_{2D})$ e quella reale in input E_{3D} diventa la mappa di anomalia Ψ_{3D}, e analogamente Ψ_{2D}. Le due mappe vengono infine aggregate in una singola anomaly map Ψ tramite **prodotto pixel-wise** ($\Psi = \Psi_{2D} \cdot \Psi_{3D}$), richiedendo che entrambe le modalità concordino nel segnalare un'anomalia

Vantaggi e Limitazioni. L'assenza di memory bank rende CFM notevolmente più veloce rispetto a M3DM e BTF, pur ottenendo performance superiori su MVTec 3D-AD. Viene inoltre proposta una tecnica di *layer pruning* sui backbone frozen per ricavare varianti compatte (Small, Tiny) del framework con

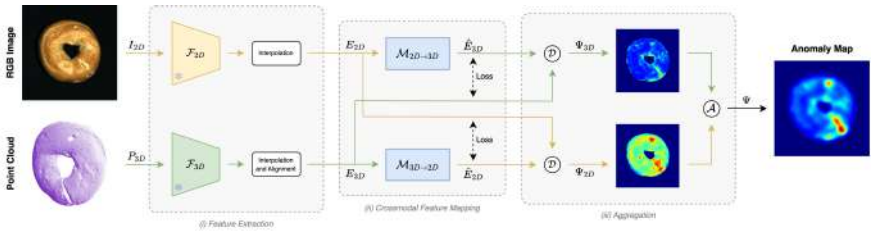

Figure 3.25: Pipeline di CFM. Date un'immagine RGB I_{2D} e una Point Cloud P_{3D}: una coppia di feature extractor $\mathcal{F}_{2D}$, $\mathcal{F}_{3D}$ basati su architetture Transformer estrae feature map pixel-allineate E_{2D}, E_{3D}. Una coppia di mapping crossmodali $\mathcal{M}_{2D\to3D}$, $\mathcal{M}_{3D\to2D}$ mappa le feature estratte da una modalità verso l'altra, elaborando ogni pixel in modo indipendente. Infine, le feature estratte E_{2D}, E_{3D} e mappate $\hat{E}_{3D}$, $\hat{E}_{2D}$ vengono confrontate tramite una funzione di discrepanza $\mathcal{D}$, producendo mappe di anomalia per modalità Ψ_{2D}, Ψ_{3D}, che vengono combinate da una funzione di aggregazione $\mathcal{A}$ per ottenere la mappa di anomalia finale Ψ. Fonte: [123]

ulteriori guadagni di efficienza. La limitazione principale è la natura intrinsecamente multimodale del metodo: a differenza di approcci basati su memory bank, CFM richiede la presenza di entrambe le modalità sia in training che in test, e non può essere applicato a scenari unimodali 2D o 3D. CFM rappresenta un contributo metodologico rilevante perché dimostra che la relazione inter-modale stessa, e non la densità della memory bank, è la vera sorgente informativa per l'anomaly detection multimodale.

3.3.4 Sviluppi Recenti e Multimodalità Avanzata (2024–2025)

Le tendenze più recenti nella ricerca 3D si muovono verso scenari ancora più complessi: multi-vista, acquisizioni in ambienti non controllati, e l'integrazione di una terza modalità (pseudo-3D da fotometria stereo), come testimoniato dai dataset Real-IAD D^3 [3] e SiM3D [4].

D^3M — D^3-Memory (2025). D^3M (*D^3-Memory*) [3] è proposto insieme al benchmark **Real-IAD D^3** (discusso in dettaglio nel Capitolo 4.5.4), che affianca alle classiche modalità RGB e point cloud una terza modalità inedita: dati **pseudo-3D** ottenuti tramite stereo fotometrico. La motivazione è che le immagini 2D non catturano difetti geometrici sottili (graffi, ammaccature), mentre i point cloud mancano di risoluzione per difetti superficiali finissimi: la pseudo-3D colma parzialmente questo divario, fornendo mappe di normali di superficie a basso costo computazionale, senza richiedere sensori di profondità dedicati.

Architettura di D^3M. D^3M estende il framework di M3DM [121] a tre modalità, introducendo due innovazioni principali (Figura 3.26):

- **Channel-Spatial Swapping (CSS)**: modulo di fusione leggero che scambia il 10% dell'informazione tra le feature map RGB X_{2D} e pseudo-3D X_{PS}:

$$X_{2D}^{\text{swap}} = (1-\alpha)\cdot X_{2D} + \alpha \cdot X_{PS}^{c\leftrightarrow s}, \quad X_{PS}^{\text{swap}} = (1-\alpha)\cdot X_{PS} + \alpha \cdot X_{2D}^{c\leftrightarrow s} \tag{3.18}$$

  dove $\alpha = 0.1$ è il rapporto di scambio e $c \leftrightarrow s$ indica lo scambio di canali e regioni spaziali $k \times k$. Lo scambio arricchisce ciascuna modalità dell'informazione dell'altra senza distruggerla, sfruttando la complementarità tra texture RGB e normali pseudo-3D

- **Fusione Contrastiva Non Supervisionata** tra le feature 2D arricchite e quelle del point cloud 3D, analoga alla contrastive loss di M3DM [121]. Opera a livello di patch confrontando feature della stessa posizione spaziale e genera una rappresentazione fusa $\mathcal{M}_{D^3}$ da memorizzare in una terza memory bank. Le feature estratte sono: DINO (ViT-b/8) [124] per RGB e pseudo-3D, PointMAE [112] per il point cloud

Tre Memory Bank e Decision Layer. Analogamente a M3DM, le tre sorgenti di feature (X_{2D}^{swap}, X_{PS}^{swap}, $\mathcal{M}_{D^3}$) creano tre memory bank separate $\mathcal{M}_{2D}$, $\mathcal{M}_{PS}$, $\mathcal{M}_D$ fusa, costruite tramite coreset subsampling. L'anomaly score finale e la segmentation map vengono prodotti da due classificatori OCSVM $\mathcal{D}_a$ e $\mathcal{D}_s$ che integrano i contributi delle tre memory bank:

$$a = \mathcal{D}_a\big(\phi(\mathcal{M}_{2D}, f_{2D}),\ \phi(\mathcal{M}_{PS}, f_{PS}),\ \psi(\mathcal{M}_{D^3}, f_{D^3})\big) \tag{3.19}$$

$$S = \mathcal{D}_s\big(\psi(\mathcal{M}_{2D}, f_{2D}),\ \psi(\mathcal{M}_{PS}, f_{PS}),\ \psi(\mathcal{M}_{D^3}, f_{D^3})\big) \tag{3.20}$$

Risultati e Contributo Principale. Su Real-IAD D^3, D^3M supera consistentemente le configurazioni monomodale (solo RGB o solo 3D) e bimodale (RGB+3D, incluso M3DM), raggiungendo un I-AUROC medio di 0.890 e un P-AUROC di 0.937, contro 0.841 e 0.922 di M3DM. L'aggiunta di dati pseudo-3D porta un contributo aggiuntivo rispetto alle due modalità classiche: in particolare, risulta indispensabile per difetti superficiali sottili che i point cloud non risolvono geometricamente ma che alterano le normali di superficie. Il contributo metodologico principale non è tanto l'architettura in sé (che riprende M3DM), quanto la dimostrazione che una modalità intermedia a basso costo acquisitivo può migliorare la detection senza sensori aggiuntivi dedicati.

3.3.5 Confronto

Analogamente a quanto fatto per i metodi 2D, presentiamo un confronto sintetico dei metodi 3D e multimodali analizzati, riportando le performance sui principali benchmark di riferimento. La molteplicità dei benchmark adottati in letteratura (MVTec 3D-AD, Real3D-AD, Real-IAD D^3) riflette la novità del campo rispetto al 2D: a differenza di MVTec AD, non esiste ancora un unico benchmark universalmente adottato per i metodi 3D.

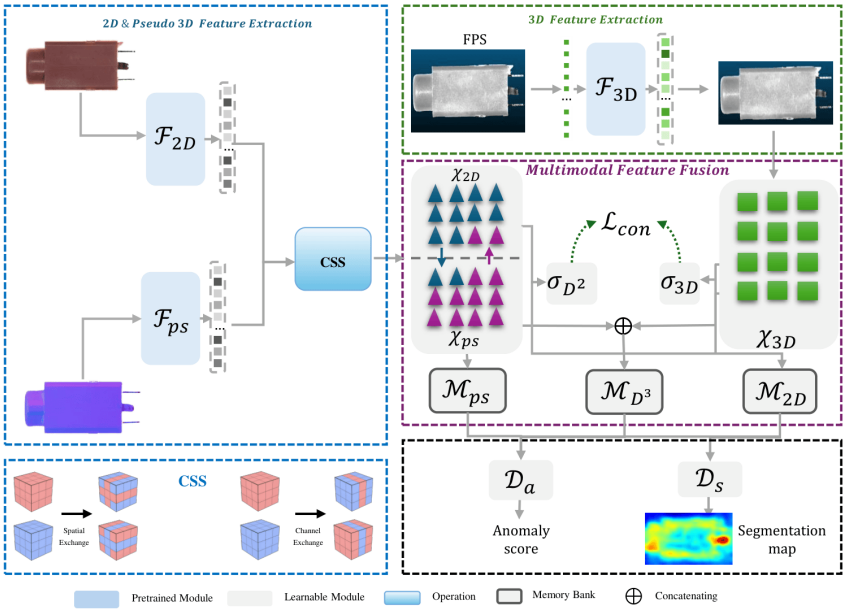

Figure 3.26: Architettura del framework D^{3}M. Il blocco in alto a sinistra illustra l'estrazione di feature 2D e pseudo-3D tramite i backbone $\mathcal{F}_{2D}$ e $\mathcal{F}_{ps}$, seguite dal modulo CSS (dettagliato in basso a sinistra) che esegue Spatial Exchange e Channel Exchange tra le due modalità. Il blocco in alto a destra mostra l'estrazione di feature 3D tramite $\mathcal{F}_{3D}$. Il blocco centrale (Multimodal Feature Fusion) fonde le tre sorgenti tramite contrastive loss $\mathcal{L}_{con}$, producendo tre memory bank $\mathcal{M}_{ps}$, $\mathcal{M}_{D^3}$, $\mathcal{M}_{2D}$. I classificatori $\mathcal{D}_a$ e $\mathcal{D}_s$ producono rispettivamente l'anomaly score e la segmentation map finale. Fonte: [3]

Table 3.4: Confronto sintetico dei principali metodi 3D e multimodali (valori indicativi)

Metodo	Anno	I-AUROC	AUPRO	Modalità	Tr.	Vel.
Benchmark: MVTec 3D-AD						
Voxel f-AnoGAN†	2021	—	0.639	RGB+3D	Sì	Lenta
BTF	2023	—	0.964	RGB+3D	No	Media
AST	2022	0.862	0.985	RGB+depth	Sì	Veloce
Shape-Guided	2023	—	0.976	RGB+3D	No*	Media
M3DM	2023	0.944	0.988	RGB+3D	No*	Lenta
CFM	2024	0.949	0.989	RGB+3D	Sì	Media
Benchmark: Real3D-AD (few-shot, 4 prototipi)						
Reg3D-AD	2023	0.704	—	3D only	No*	Media
Benchmark: Real-IAD D³						
M3DM‡	2023	0.841	0.922	RGB+3D	No*	Lenta
D^3M	2025	0.890	0.937	RGB+3D+PS	No*	Lenta

Note: Valori in proporzione (0–1). †Voxel f-AnoGAN: AUPRO 0.583 in configurazione solo-3D. *Costruzione offline di memory bank o coreset richiesta. ‡M3DM riportato come baseline su Real-IAD D^3 nel paper di D^3M. PS = dati pseudo-3D da fotometria stereo. Velocità relativa su GPU moderna: Veloce (<50ms), Media (50-200ms), Lenta (>200ms). I-AUROC non disponibile (—) per metodi che riportano esclusivamente metriche di segmentazione.

Guida alla Selezione per Scenario Applicativo (3D)

Tendenze Evolutive e Considerazioni Finali (Metodi 3D)

L'analisi comparativa dei metodi 3D rivela un percorso evolutivo più rapido e compresso rispetto al 2D: in soli quattro anni (2021–2025) il campo è passato da semplici adattamenti voxel, con performance modeste, a metodi multimodali avanzati competitivi con i migliori sistemi 2D. Cinque tendenze principali caratterizzano questa evoluzione:

1. **Da voxel a point cloud**: I primi metodi voxel (Voxel AE, Voxel f-AnoGAN) hanno rapidamente ceduto il passo a rappresentazioni native per dati 3D. L'adozione di descrittori geometrici rotation-invariant (FPFH in BTF) e di backbone pre-addestrati su point cloud (PointMAE in M3DM, Reg3D-AD) ha permesso miglioramenti di performance significativi.

2. **Multimodalità**: I metodi unimodali 3D raggiungono prestazioni inferiori rispetto alle alternative multimodali RGB+3D. La complementarità delle due sorgenti (colore e texture dall'immagine, geometria e profondità dal point cloud) è un principio consolidato, come dimostrato da AST, M3DM e CFM.

3. **Memory bank vs. mapping crossmodale**: Emerge una separazione nei metodi, tra approcci che memorizzano esplicitamente feature normali

Table 3.5: Guida pratica alla selezione del metodo 3D/multimodale

Scenario Prioritario	Metodo Consigliato e Motivazione
Accuratezza Massima (3D)	**CFM** (crossmodal mapping, superiore a M3DM su MVTec 3D-AD) o **D^{3}M** se disponibile la terza modalità pseudo-3D.
Efficienza Computazionale	**BTF** (descrittori FPFH, nessun training) o **AST** (student-teacher leggero) per il miglior compromesso accuratezza/velocità.
Dati Limitati (Few-Shot)	**Reg3D-AD** per scenari con pochissimi prototipi (≤ 4), grazie a RANSAC che compensa la scarsità di esempi normali.
Solo Point Cloud (no RGB)	**BTF** (FPFH + k-NN, rotation-invariant) come baseline robusto. **Shape-Guided** per scenari che richiedono anche localizzazione precisa.
Fusione RGB+3D Completa	**M3DM** o **CFM** per texture/geometria che si completano; **D^{3}M** se si dispone di fotometria stereo.
Deployment Rapido (no training)	**BTF** o **Shape-Guided**: sufficiente costruzione offline della memory bank, nessun addestramento di reti richiesto.

in memory bank (M3DM, Shape-Guided) e approcci che apprendono relazioni inter-modali (CFM). Il secondo approccio offre un percorso più efficiente, eliminando la necessità di strutture di ricerca approssimata su milioni di feature.

4. **Oltre RGB+3D**: La terza modalità pseudo-3D introdotta da D^{3}M apre la strada a framework a n modalità, dove ciascuna sorgente copre un range diverso di difetti (superficiali, geometrici, fotometrici). Questo suggerisce possibili future architetture ancora più flessibili nel numero e nel tipo di input.

5. **Benchmark ancora frammentati**: A differenza del 2D, dove MVTec AD ha rappresentato un punto di convergenza univoco, il panorama 3D è ancora diviso tra MVTec 3D-AD, Real3D-AD e Real-IAD D^3. La mancanza di un benchmark condiviso rende i confronti diretti tra metodi più difficili e meno conclusivi.

Conclusioni Generali. Il campo dell'anomaly detection industriale ha percorso in meno di un decennio un'evoluzione straordinaria, che ha portato i metodi 2D a saturare i principali benchmark e ha aperto un nuovo fronte di ricerca nella tridimensionalità. Oggi la frontiera si muove verso l'integrazione intelligente di modalità complementari, la generalizzazione a nuovi domini con pochissimi esempi, e la compatibilità con vincoli hardware stringenti. I metodi 3D e multimodali non sostituiscono i metodi 2D, ma li completano: la scelta tra approcci visivi, geometrici o ibridi rimane strettamente legata ai sensori disponibili, alla natura dei difetti attesi e ai requisiti operativi del contesto produttivo specifico.

Chapter 4

Dataset

4.1 Il Ruolo dei Dataset

Lo sviluppo e il progresso di qualsiasi campo del *deep learning* sono strettamente legati alla disponibilità di **dataset** di alta qualità, diversificati e rappresentativi della complessità degli scenari reali. I dataset non costituiscono semplici collezioni di immagini o campioni, ma rappresentano la base per definire problemi, formulare ipotesi scientifiche, addestrare e validare gli algoritmi, misurare i progressi ottenuti. L'efficiacia dei metodi può essere valutata solo grazie a benchmark condivisi e rigorosi.

Nel contesto specifico dell'**anomaly detection** per applicazioni industriali, i dataset assumono un'importanza ancora più critica. A differenza di altri domini dove i dati possono essere raccolti in modo relativamente scalabile (ad esempio, attraverso il *web scraping* o l'acquisizione da sensori diffusi), le immagini di prodotti industriali, specialmente quelle che riportano difetti difficili da riprodurre, richiedono acquisizione controllata e spesso condotta in ambienti produttivi reali. Inoltre, la natura stessa dei difetti industriali (variabili da micro-graffi superficiali a difetti strutturali complessi, da anomalie di texture ad anomalie logiche) impone che i dataset siano progettati con cura per coprire uno spettro ampio e realistico di possibili scenari.

Un dataset di qualità per l'anomaly detection industriale deve soddisfare alcuni requisiti fondamentali [1], [66]:

- **Immagini ad alta risoluzione**: La rilevazione di micro-difetti (graffi sottili, piccole contaminazioni) richiede immagini con sufficiente dettaglio spaziale per preservare le caratteristiche distintive delle anomalie.
- **Annotazioni pixel-precise**: Per applicazioni industriali che richiedono localizzazione accurata dei difetti, sono necessarie maschere di *ground truth* con segmentazioni precise dei confini delle anomalie.
- **Variabilità controllata**: I dati normali devono presentare variabilità

intra-classe sufficiente (variazioni di illuminazione, pose, texture) per addestrare modelli robusti, senza compromettere la definizione di normalità.

- **Diversità di anomalie**: Le tipologie di difetti devono coprire un ampio spettro di manifestazioni visive: difetti strutturali (crepe, denti), difetti superficiali (graffi, macchie), anomalie logiche (componenti mancanti o disallineati).
- **Scalabilità**: I dataset devono essere sufficientemente grandi per permettere l'addestramento di modelli *deep learning* e una validazione statistica robusta dei risultati.

L'evoluzione dei metodi di anomaly detection, illustrata in 3.2, è andata di pari passo con l'evoluzione dei dataset a disposizione. I primi approcci, spesso basati su **autoencoder** o **GAN**, sono stati valutati su dataset relativamente piccoli e omogenei (ad esempio, immagini in scala di grigi di singole componenti). La pubblicazione di dataset come **MVTec AD** ha segnato una svolta, offrendo una varietà di oggetti e texture, annotazioni pixel-precise e una chiara separazione tra training (solo esempi normali) e test (con anomalie). Questo ha permesso la nascita e il confronto di metodi più sofisticati, basati su *feature pre-addestrate*, *memory bank* e *normalizing flows*.

Più recentemente, la necessità di affrontare scenari industriali sempre più complessi (ispezioni 3D, illuminazione variabile, oggetti multi-vista) ha portato verso la creazione di dataset **multimodali** (RGB, depth, point cloud) e **large-scale**, spesso acquisiti in ambienti reali o generati sinteticamente con alto realismo. Questi dataset consentono di testare la robustezza e la generalizzazione dei metodi esistenti, ma portano anche verso lo sviluppo di nuove architetture in grado di elaborare informazioni visive e geometriche in modo integrato.

In questo capitolo, presenteremo una panoramica strutturata dei principali dataset utilizzati nella letteratura di *anomaly detection* industriale. Dopo averne analizzato le caratteristiche distintive e le metriche di valutazione associate, discuteremo come l'evoluzione dei dataset abbia influenzato lo sviluppo di algoritmi sempre più accurati, robusti e pronti per il deployment in scenari industriali reali.

4.2 Evoluzione dei Dataset (2018–2025)

L'evoluzione dei dataset per l'anomaly detection industriale riflette la crescente complessità delle applicazioni industriali e l'emergere di nuove esigenze. Possiamo identificare tre fasi principali, ciascuna caratterizzata da un salto in termini di modalità di acquisizione, tipologia di anomalie e scenario valutativo.

4.2.1 Prima generazione (2018–2020): Benchmark 2D e definizione del task.

I primi dataset, come **Magnetic Tile Defects** (2018) [125] e **MVTec AD** (2019) [66] si concentrarono su immagini 2D (scala di grigi o RGB) di oggetti

isolati o texture, con annotazioni pixel-precise per difetti superficiali (graffi, crepe, macchie). MVTec AD in particolare stabilì lo standard per dataset 2D RGB con annotazioni pixel-level, diventando il benchmark di riferimento universale.

4.2.2 Seconda generazione (2021–2022): Espansione multimodale e anomalie complesse.

Per testare modelli più avanzati e affrontare limiti dei dati puramente visivi, emersero dataset che incorporano informazioni 3D e anomalie di tipo logico-strutturale. **MVTec 3D-AD** (2021) [2] introdusse depth map e point cloud insieme alle immagini RGB. **MVTec LOCO AD** (2022) [1] focalizzò l'attenzione sulle *logical anomalies* (componenti mancanti, assemblaggi errati), richiedendo ai modelli di ragionare sulla struttura globale dell'oggetto. Parallelamente, dataset sintetici come **Eyecandies** (2022) [126] permisero un controllo fine su illuminazione, materiali e geometrie, facilitando la validazione in condizioni perfettamente controllate.

4.2.3 Terza generazione (2023–2025): Scenari realistici, multi-vista e large-scale.

I dataset più recenti hanno spostato l'attenzione verso condizioni realistiche. Dataset come **Real-IAD** (2024) [127] e **RAD** (2024) [128] offrono migliaia di immagini acquisite da più viewpoint, con oggetti riflettenti, trasparenti e in condizioni di illuminazione variabile. **MVTec AD 2** (2025) [6] introduce sfide aggiuntive come difetti ai bordi dell'immagine e variazione nel numero di oggetti per scena. L'obiettivo è ridurre il gap tra performance su benchmark controllati e deployment industriale reale.

La Tabella 4.1 sintetizza questa progressione temporale, evidenziando l'incremento di complessità e realismo.

4.2.4 Metriche di Valutazione: Panoramica

La valutazione rigorosa degli algoritmi di anomaly detection richiede metriche standardizzate che catturino aspetti diversi della performance: accuratezza a livello di immagine, precisione nella localizzazione, robustezza ai falsi positivi, etc. Le metriche più utilizzate nei benchmark industriali possono essere classificate in due categorie principali: *threshold-free* (indipendenti da una soglia di decisione) e *threshold-dependent* (che richiedono la binarizzazione dello score di anomalia).

Metriche Threshold-Free.

- **AUROC (Area Under the Receiver Operating Characteristic Curve)**: misura la capacità del modello di separare classi normali e anomale al variare della soglia. Viene riportata sia a livello immagine

Table 4.1: Evoluzione dei dataset per anomaly detection industriale (2018-2025). La suddivisione in tre generazioni riflette l'espansione dalle basi 2D alla multimodalità, fino agli scenari realistici e multi-vista.

Anno	Dataset	Contributi Principali / Innovazioni
	Prima Generazione: Benchmark 2D e definizione del task (2018–2020)	
2018	Magnetic Tile Defects	Dataset pionieristico per difetti industriali; immagini in scala di grigi; valutazione di modelli di saliency detection.
2019	**MVTec AD**	**Benchmark di riferimento universale**; 15 categorie (oggetti e texture); annotazioni pixel-precise; separazione chiara training/test; standardizzazione di metriche (AUROC, PRO).
2021	MPDD	Focus su parti metalliche; scenari tipici dell'industria metallurgica (graffi, fori, ruggine).
2022	VisA	Oggetti con strutture complesse e multiple istanze; introduce protocolli few-shot e high-shot per valutare la sensibilità locale dei metodi self-supervised.
	Seconda Generazione: Espansione multimodale e anomalie complesse (2021–2022)	
2021	MVTec 3D-AD	Primo dataset 3D industriale; unisce RGB, depth map e point cloud; anomalie geometriche (graffi, ammaccature, contaminazioni).
2022	MVTec LOCO AD	Introduzione delle **anomalie logiche** (errori di assemblaggio, componenti fuori posto); metrica sPRO per valutare il rilevamento di difetti contestuali.
2022	Eyecandies	Dataset **sintetico** su larga scala (90K campioni); controllo completo su illuminazione, materiali e anomalie; include RGB, depth e mappe delle normali.
	Terza Generazione: Scenari realistici, multi-vista e large-scale (2023–2025)	
2023	Real3D-AD	Dataset 3D ad alta precisione; setting **few-shot** (4 campioni per categoria); copertura a 360° senza punti ciechi.
2023	PAD / MAD	Setting **pose-agnostic**; acquisizioni multi-vista (∼20 viewpoint); versioni sia reali (MAD-Real) che sintetiche (MAD-Sim).
2024	Real-IAD	Dataset **large-scale** (150K immagini); 30 categorie industriali; acquisizione multi-vista (5 angolazioni); progettato per il deployment in scenari reali.
2024	RAD	**Multi-vista estremo** (60+ viewpoint); setting **noisy-pose** per testare la robustezza; include ricostruzione 3D a partire dalle viste RGB.
2025	**MVTec AD 2**	Introduce **sfide realistiche**: oggetti trasparenti/riflettenti, condizioni di illuminazione variabili, anomalie ai bordi dell'immagine, variazione nel numero di oggetti.
2025	Real-IAD D^3	Estensione **tri-modale** di Real-IAD: RGB, point cloud e mappe delle normali; 20 categorie; 69 gruppi di difetti per valutazione fine.
2025	SiM3D	Dataset **ibrido** (reale + sintetico); multi-vista (12–36 viste per istanza); valutazione voxel-level per segmentazione 3D.

(*Image AUROC*) che a livello pixel (*Pixel AUROC*). Valori prossimi a 1 indicano prestazioni eccellenti. È la metrica più comunemente riportata per la sua robustezza a classi sbilanciate.

- **PRO (Per-Region Overlap)** [1]: progettata per valutare la qualità della localizzazione, specialmente per difetti piccoli. Calcola la curva Precision-Recall media su ogni regione anomala indipendentemente dalla sua dimensione, evitando che difetti grandi dominino la metrica. Il suo integrale (AUPRO) è diventato standard per dataset con annotazioni pixel-precise.

Metriche Threshold-Dependent.

- **F1-Score**: media armonica di precision e recall, calcolata dopo aver applicato una soglia ottimale (tipicamente sul validation set). Sensibile allo sbilanciamento delle classi, ma utile per scenari dove è critico bilanciare falsi positivi e falsi negativi.
- **MAE (Mean Absolute Error)**: utilizzata principalmente in metodi basati su ricostruzione o saliency map, misura l'errore medio assoluto tra la mappa predetta e la ground truth.

La scelta della metrica influenza direttamente la comparabilità dei risultati. Dataset incentrati sulla localizzazione (es. MVTec AD) privilegiano AUROC e PRO, mentre dataset con anomalie logiche (es. MVTec LOCO AD) possono introdurre metriche ad-hoc come *sPRO* (Saturated PRO). Nei paragrafi seguenti, per ciascun dataset indicheremo le metriche con cui è stato principalmente valutato, collegandole alle specifiche esigenze applicative che lo hanno ispirato.

4.3 Dataset 2D RGB

I dataset basati su immagini RGB 2D rappresentano il nucleo della ricerca in anomaly detection visiva. Questa sezione presenta i principali benchmark ordinati cronologicamente, evidenziando le caratteristiche distintive e i contributi di ciascuno.

4.3.1 Surface Defect Saliency of Magnetic Tile (2018)

Rappresentante della prima generazione di dataset, Magnetic Tile [125] è stato uno dei primi dataset specificamente creati per anomaly detection industriale, focalizzato sul rilevamento di difetti superficiali in piastrelle magnetiche.

Caratteristiche. Il dataset contiene **1,344 immagini in scala di grigi** di piastrelle magnetiche distribuite in 6 classi a seconda del tipo di difetto: Blowhole, Crack, Fray, Break, Uneven, e Free (assenza di difetti). Le immagini catturano difetti tipici del processo di produzione di componenti magnetici (Figura 4.1).

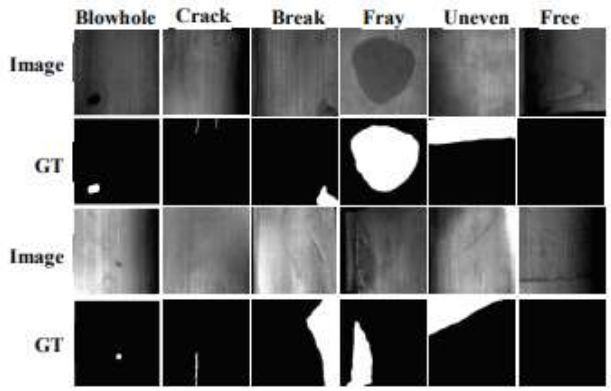

Figure 4.1: Esempi di difetti superficiali su piastrelle magnetiche con le rispettive maschere di ground truth a livello di pixel (GT) [125].

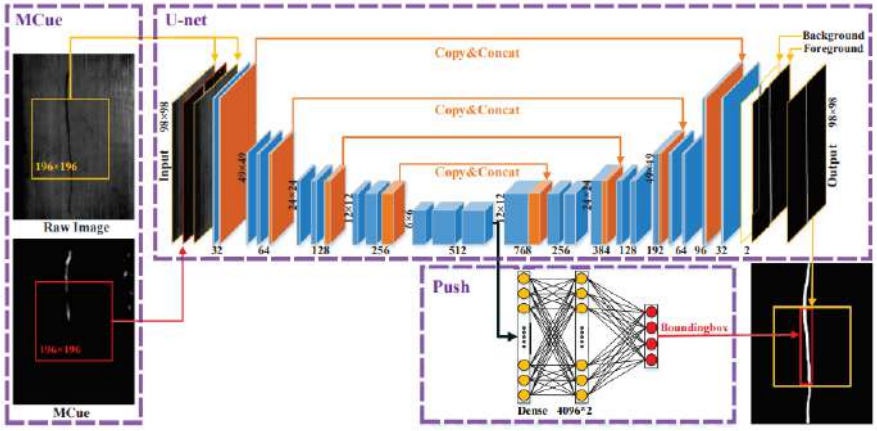

Figure 4.2: Schema dell'architettura MCuePushU con i tre componenti principali: MCue, U-Net e Push Network.

Contributo Metodologico. Il dataset introdusse **MCuePushU** (Figura 4.2), un modello di saliency detection che combina:

1. Estrazione di *dominant cues* (caratteristiche salienti) dai dati
2. Fusione di questi cues in una rete U-Net [99] tramite operazioni aritmetiche sulle immagini
3. Un modulo Push Network che produce bounding boxes per evidenziare i difetti predetti

Valutazione. Gli esperimenti confrontarono MCuePushU con i principali metodi non deep-learning dell'epoca su dataset di scene naturali: GMR [129], MBP [130], MC [131], RBD [132], DRFI [133]. Le metriche utilizzate furono:

- **AUC (Area Under ROC)**: misura la capacità di separare regioni normali e anomale
- **MAE (Mean Absolute Error)**: errore medio assoluto tra saliency map e ground-truth
- $\mathbf{F}_\beta$: massimo della curva Precision-Recall, con β che bilancia precision e recall.

Limitazioni. Sebbene pioneristico, il dataset presentava limitazioni significative: immagini in scala di grigi (perdita di informazioni cromatiche), dimensione limitata ($< 1,500$ immagini), focus su un singolo tipo di oggetto. Tuttavia, il suo contributo fu fondamentale nel dimostrare la necessità di dataset industriali dedicati e nel stabilire protocolli di valutazione che sarebbero stati adottati e ampliati da benchmark successivi come MVTec AD.

4.3.2 MVTec AD (2019)

MVTec Anomaly Detection (MVTec AD) [66] è diventato il benchmark di riferimento universale per l'anomaly detection industriale, utilizzato come baseline in praticamente tutti i lavori successivi.

Caratteristiche. Il dataset contiene **5,354 immagini RGB ad alta risoluzione** distribuite su 15 categorie distinte: 10 categorie di oggetti (bottle, cable, capsule, hazelnut, metal nut, pill, screw, toothbrush, transistor, zipper) e 5 categorie di texture (carpet, grid, leather, tile, wood). Le immagini normali (3629 defect-free) costituiscono il training set, mentre il test set include 1725 immagini con anomalie. Sono presenti **73 tipologie di difetti** generati manualmente (in media 5 per categoria), riproducendo difetti realistici: graffi, denti, contaminazioni, deformazioni, componenti mancanti, disallineamenti (Figura 4.3).

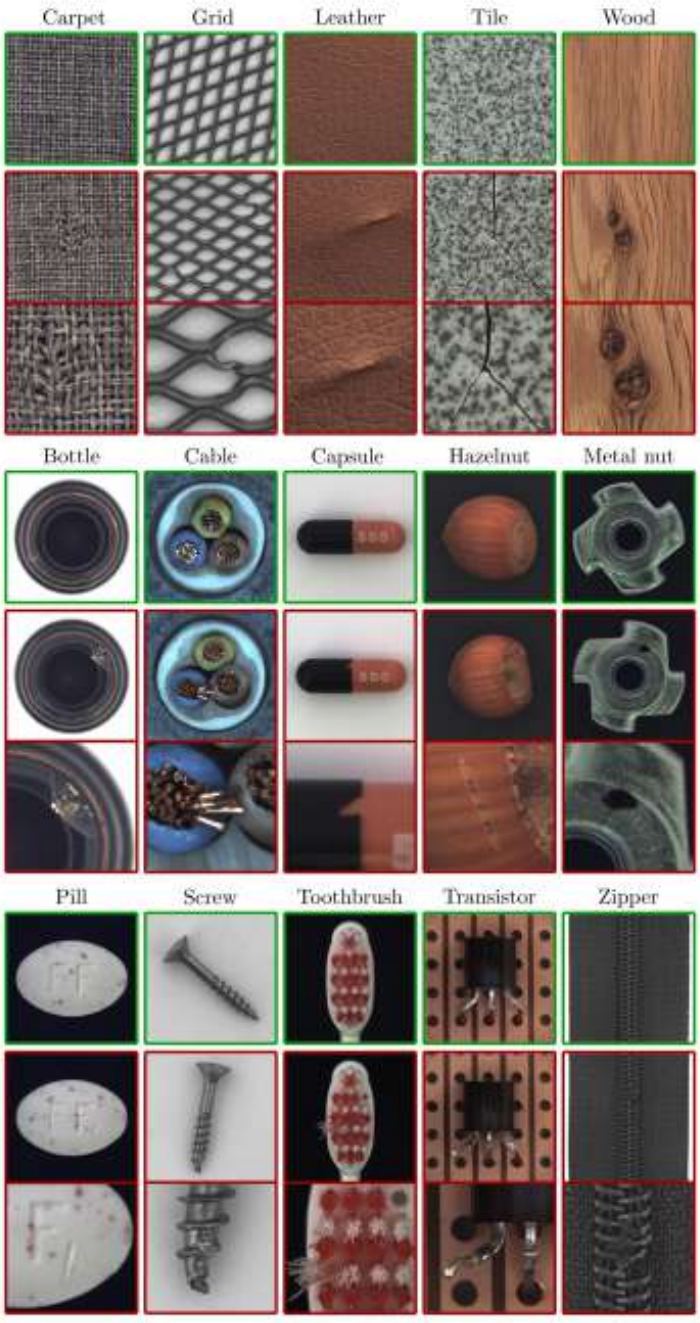

Figure 4.3: Esempi di immagini per le cinque categorie di texture e le dieci categorie di oggetti del dataset MVTec AD [66]. Per ogni categoria, la riga superiore mostra un'immagine priva di anomalie, la riga centrale un esempio anomalo e la riga inferiore un ingrandimento che evidenzia la regione difettosa.

Annotazioni Pixel-Precise. Ogni immagine anomala è corredata da una *ground truth* mask binaria che segmenta a livello di pixel la regione difettosa. Questo permette la valutazione sia del task di **image-level classification** (l'immagine è normale o anomala?) che di **pixel-level localization** (dove si trova esattamente il difetto?).

Protocollo Sperimentale.

- **Training**: solo su immagini normali (semi-supervised/one-class)
- **Testing**: su immagini normali e anomale, con valutazione separata per detection e localization
- **Data augmentation**: random crop di patch rettangolari ruotate per le texture, mentre per gli oggetti si applicano traslazioni, rotazioni casuali e mirroring dove possibile, generando 10000 patch di training per categoria.

Metodi Valutati e Performance Iniziali. Il paper originale valutò diversi approcci rappresentativi dell'epoca, rivelando limiti significativi:

- **AnoGAN** [84]: GAN-based, discusso in Sezione 3.2.1. 70% image AUROC, in difficoltà nelle categorie che presentano oggetti con più variazioni (forma, orientamento).
- **Autoencoder** (MSE/SSIM): varianti con loss L2 (MSE) e SSIM (Structural Similarity), discussi in Sezione 3.2.2. 80-85% image AUROC (problemi di false positive, falliscono nel modellare piccoli dettagli).
- **CNN Feature Dictionary**: metodo basato su feature pre-addestrate, performance migliori ma limitate
- **Texture Inspection GMM-Based**: efficace per texture ma non per oggetti
- **Variation Model**: adatto a oggetti con variazioni limitate

Metriche Standardizzate. Il dataset introdusse due metriche fondamentali:

- **AUROC (Area Under ROC Curve)**: per valutazione a livello immagine e pixel
- **PRO (Per-Region Overlap)**: metrica pixel-level che penalizza errori su piccole regioni anomale, affrontando il bias dell'AUROC pixel-level verso grandi regioni (discussa in Sezione 4.6.2)

Impatto. MVTec AD ha avuto un profondo impatto sulla ricerca:

1. Definì uno standard riproducibile e condiviso per la valutazione
2. Espose i limiti dei metodi contemporanei (GAN-based, autoencoder)
3. Motivò lo sviluppo di approcci più avanzati che oggi superano il 99% AUROC
4. Divenne la piattaforma di riferimento per confronti oggettivi e riproducibili

Disponibilità e Adozione. Distribuito sotto licenza Creative Commons, MVTec AD è ampiamente utilizzato sia in ricerca che in contesti industriali, con implementazioni integrate nei principali framework di deep learning e continui aggiornamenti (come MVTec AD 2, 2025).

4.3.3 MPDD (2021)

Metal Parts Defect Dataset (MPDD) [134] fu introdotto per colmare una lacuna specifica: la scarsità di dataset focalizzati su parti metalliche, un dominio con caratteristiche distintive (superfici riflettenti, texture uniformi, variazioni cromatiche sottili).

Caratteristiche. MPDD contiene **1,346 immagini RGB** distribuite su 6 classi di componenti metallici: bracket black, bracket brown, bracket white, connector, metal plate, tubes (Figura 4.4). Il dataset è suddiviso in:

- **Training set**: 888 campioni (solo normali)
- **Test set**: 458 campioni (176 normali, 282 anomali)

Le anomalie coprono scenari tipici: scratches (graffi), holes (buchi), missing parts (parti mancanti), rust (ruggine). Ogni immagine anomala è annotata con una pixel-precise ground truth mask.

Metodi Valutati. MPDD confrontò due categorie di metodi:

- **Feature extraction-based**: PatchCore [81], CFLOW-AD [93], PaDiM [82], Semi-Orthogonal [135], SPADE [92]
- **Reconstruction-based**: DAGAN [136], Skip-GANomaly [87], ITAE [137]

Gap di Performance. I risultati mostrarono la superiorità degli approcci feature-based (PatchCore, PaDiM) rispetto ai reconstruction-based. Tuttavia, i metodi subivano un **calo di performance** su MPDD:

- **Differenza AUROC image-level**: fino a **23.12%** per singoli metodi
- **Differenza media**: **15.24%** tra le performance aggregate sui due dataset

Questo gap dimostrò in modo empirico che la vera generalizzazione richiede test su dataset che riflettano le condizioni operative reali.

Figure 4.4: Campioni del dataset MPDD [134]: nella riga superiore gli esempi normali del training set, nelle righe centrale e inferiore i campioni del test set con le rispettive anomalie evidenziate.

Conclusione e Impatto. MPDD contribuì a spostare il focus della comunità dalla semplice ottimizzazione delle metriche su benchmark standardizzati verso la robustezza in condizioni reali. Il suo messaggio chiave è che per lo sviluppo e il miglioramento dei metodi di AD, è necessario testare questi metodi su dataset basati su applicazioni reali specifiche.

4.3.4 VisA / SPot-the-Difference (2022)

VisA (Visual Anomaly) [138] introdusse sfide significative e scenari più complessi: oggetti con strutture dettagliate, istanze multiple nello stesso frame, e anomalie sia superficiali che strutturali.

Caratteristiche. Il dataset contiene **10,821 immagini RGB** (9,621 normali, 1,200 anomale) di 12 oggetti organizzati in 3 domini:

1. **Complex structures**: quattro tipi di printed circuit boards (PCB), con strutture complesse contententi transistors, capacitors, chips, etc.
2. **Multiple instances**: più oggetti nella stessa immagine (Macaroni1, Macaroni2, Capsule, Candele). Le istanze di Macaroni2 e Capsules sono molto diverse in posizione e orientamento degli oggetti.
3. **Single instance**: singolo oggetto per immagine (Cashew, Chewing gum, Fryum e Pipe fryum)

Le anomalie, generate manualmente per produrre risultati realistici, coprono due categorie principali:

- **Surface defects**: scratches, dents, color spots, cracks
- **Structural defects**: misplacement (componenti disallineati), missing parts (parti mancanti)

Ci sono 5-20 immagini per tipo di anomalia e ogni immagine può contenerne più di una. Tutte le anomalie sono annotate con image-level e pixel-level labels.

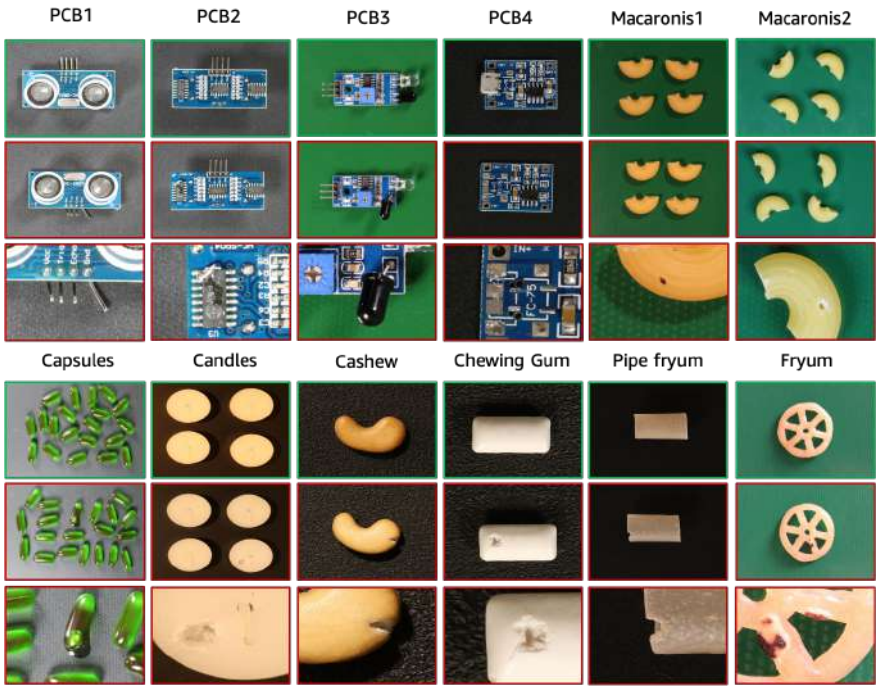

Figure 4.5: Campioni del dataset VisA. Prima riga: immagini normali; seconda riga: immagini anomale; terza riga: ingrandimento delle regioni anomale con ground truth sovrapposta.

Benchmark Protocols. VisA introdusse protocolli few-shot e high-shot in aggiunta al paradigma one-class tradizionale:

- **1-class vs 2-class training**:
  - **1-class**: addestramento su una singola classe (paradigma standard). Il 90% delle immagini normali viene assegnato come *train set* mentre il restante 10% e le immagini anomale costituiscono il *test set*.

  - **2-class**: addestramento su dati normali **e** anomali per valutare la *cross-category generalization*: i campioni anomali usati in training provengono da categorie diverse rispetto a quelle di test, misurando la capacità del modello di trasferire la nozione di anomalia tra domini distinti.

- **High-shot vs Low-shot** (definiti da percentuali di split):
  - **High-shot**: 60% dei dati normali/anomali per training, 40% per test
  - **Low-shot**: 20% dei dati normali/anomali per training, 80% per test
- **k-shot (k=5,10)** (campionamento assoluto dal Low-shot):
  - Dal training set Low-shot, si campionano **k immagini** dalla classe normale e **k immagini** dalla classe anomala
  - Totale training: **2k immagini** (es: 10 immagini totali per 5-shot)
  - Le performance sono mediate su 5 run random per robustezza statistica

SPot-the-Difference (SPD). Il paper introdusse SPD, un metodo di self-supervised pre-training progettato per aumentare la sensibilità locale di metodi SSL (Self-Supervised Learning) come SimCLR [139], MoCo [140], SimSiam [141].

SPD utilizza **SmoothBlend**, una tecnica di local augmentation che genera pseudo-anomalie mescolando patch da immagini diverse con smoothing ai bordi per creare transizioni realistiche. Durante il pre-training, il modello impara a rilevare queste perturbazioni locali, sviluppando una sensibilità utile per l'anomaly detection.

Metodi Valutati. Confronto con patch-based methods come PatchCore [81] e PaDiM [82], dimostrando che SPD migliora le performance nei regimi few-shot.

Metriche. Oltre all'AUROC standard, VisA utilizzò **AUPR (Area Under Precision-Recall curve)**, utile in scenari sbilanciati dove le anomalie sono rare.

Contributo. VisA sottolineò l'importanza di:

1. **Scenari multi-istanza**: più comuni rispetto a single-object
2. **Efficienza dati**: few-shot learning per ridurre i costi di acquisizione
3. **Pre-training self-supervised**: come alternativa al fine-tuning supervisionato

4.3.5 MVTec LOCO AD (2022)

MVTec Logical Constraints Anomaly Detection (MVTec LOCO AD) [1] introdusse per la prima volta le **anomalie logiche** oltre a quelle strutturali.

Motivazione. In molti scenari industriali, un prodotto può essere strutturalmente integro ma logicamente errato. Esempi tipici possono includere:

- **Montaggio errato**: componenti montati nella posizione sbagliata, orientamento invertito
- **Violazione di vincoli**: combinazioni di colori non conformi, numero errato di elementi
- **Errori di assemblaggio**: parti mancanti in un insieme

Queste anomalie non sono rilevabili attraverso l'analisi di texture o geometria locale ma richiedono comprensione contestuale e reasoning sulle relazioni spaziali tra componenti, andando oltre il rilevamento di difetti superficiali.

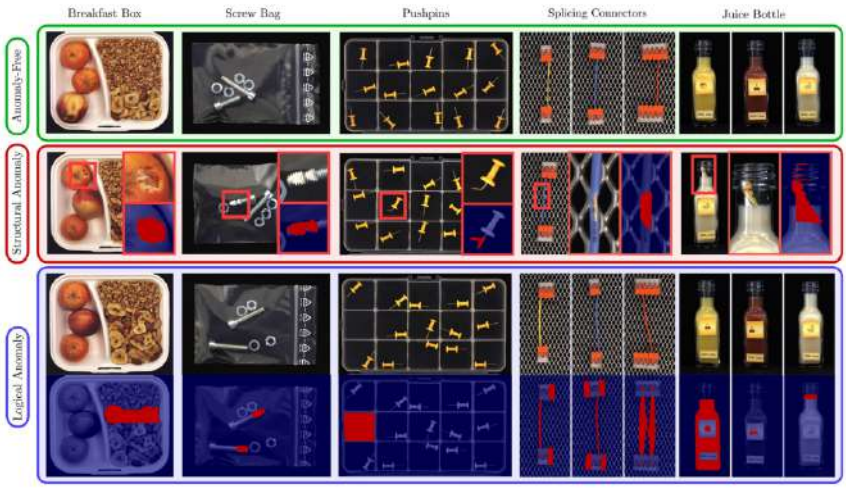

Figure 4.6: Esempi del dataset MVTec LOCO AD per ciascuna delle cinque categorie. Oltre alle immagini anomaly-free (riga superiore), il dataset include anomalie strutturali (riga centrale) e logiche (riga inferiore), tutte annotate con maschere pixel-precise.

Caratteristiche. Il dataset contiene immagini RGB di **5 categorie di oggetti** (Breakfast Box, Screw Bag, Pushpins, Splicing Connectors, Juice Bottle), ciascuna con vincoli logici specifici (Figura 4.6). Il dataset è suddiviso in:

- **Training set**: 1,772 immagini (anomaly-free)
- **Validation set**: 304 immagini (anomaly-free)
- **Test set**: 1,568 immagini (con e senza anomalie)

Le anomalie sono sia **strutturali** (difetti fisici) che **logiche** (violazioni di vincoli logici). Entrambi i tipi hanno annotazioni pixel-precise.

Metodi Valutati. Benchmark di diversi approcci:

- **Autoencoders**: Deterministic (AE), Variational (VAE) [88], Memory-guided (MNAD) [91]
- **GAN-based**: f-AnoGAN [86]
- **Feature-based**: SPADE [92], Student-Teacher [79]
- **GCAD (Global Context Anomaly Detection)**: metodo specificamente progettato per anomalie logiche

Metriche. Oltre all'AUROC, LOCO AD introdusse **sPRO (saturated Per-Region Overlap)**, una generalizzazione della metrica PRO. Mentre PRO standard calcola la media dell'overlap (percentuale di pixel correttamente predetti) per ciascuna regione anomala, pesando tutte le regioni allo stesso modo indipendentemente dalla loro dimensione, **sPRO** satura il contributo di una regione una volta che l'overlap supera una certa soglia.

Questo approccio è particolarmente adatto per anomalie logiche: ad esempio, per un componente mancante (come una puntina da disegno assente in un contenitore), è sufficiente che il metodo segmenti un'area delle dimensioni di una puntina all'interno dello scomparto vuoto, senza bisogno di segmentare perfettamente tutti i pixel della regione annotata (che potrebbe essere più grande).

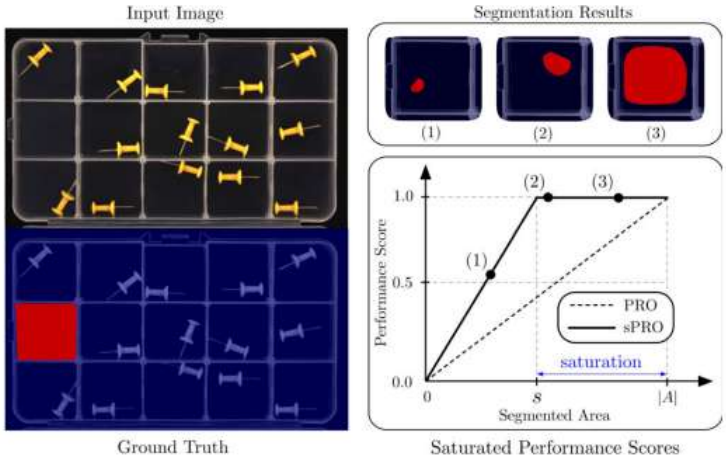

Figure 4.7: Illustrazione della metrica sPRO. Data un'anomalia annotata A e una soglia di saturazione s, una volta che l'overlap tra la regione predetta e la ground truth supera s, il task di segmentazione si considera risolto. Rispetto a PRO,sPRO evita di premiare eccessivamente segmentazioni molto estese (caso 3) rispetto a quelle già sufficientemente precise (caso 2).

Risultati Chiave. I risultati rivelarono che:

1. Metodi reconstruction-based (AE, VAE, GAN) hanno performance significativamente peggiori su anomalie logiche rispetto a strutturali
2. Approcci feature-based con reti pre-addestrate generalizzano meglio
3. Il rilevamento di anomalie logiche rimane una sfida aperta, con AUROC tipicamente 5-10% inferiore rispetto a MVTec AD standard

Impatto. MVTec LOCO AD evidenziò la necessità di metodi con capacità di reasoning contestuale, spingendo lo sviluppo di approcci basati su attention mechanisms e vision-language models capaci di comprendere relazioni semantiche.

4.3.6 Real-IAD (2024)

Real-world Industrial Anomaly Detection (Real-IAD) [127] rappresenta un passo significativo per la riduzione del gap tra performance su benchmark controllati e scenari di deployment reale.

Caratteristiche. Real-IAD è uno dei dataset più grandi disponibili:

- **150,000 immagini ad alta risoluzione**
- **30 categorie** diverse di oggetti di diversi materiali (metallo, plastica, legno, ceramica e mixed materials)
- **Acquisizione multi-view**: ogni oggetto viene catturato da 5 angolazioni distinte (dall'alto e da quattro angoli simmetrici a 45 gradi)

Questa struttura multi-view consente di valutare non solo la capacità di rilevamento ma anche la robustezza dei metodi rispetto a variazioni di prospettiva.

Protocolli di Valutazione. Real-IAD introduce due setting sperimentali:

1. Unsupervised IAD (UIAD). L'algoritmo impara pattern e strutture *solo* da campioni normali (paradigma standard). Il training set contiene solo immagini normali, mentre il testing set ha sia campioni normali che anomali. Benchmark:

- **Embedding-based**: PatchCore [81], PaDiM [82], CFlow [93]
- **Data-augmentation-based**: SimpleNet [105], DeSTSeg [142]
- **Reconstruction-based**: RD (Reverse Distillation) [102], UniAD [137]

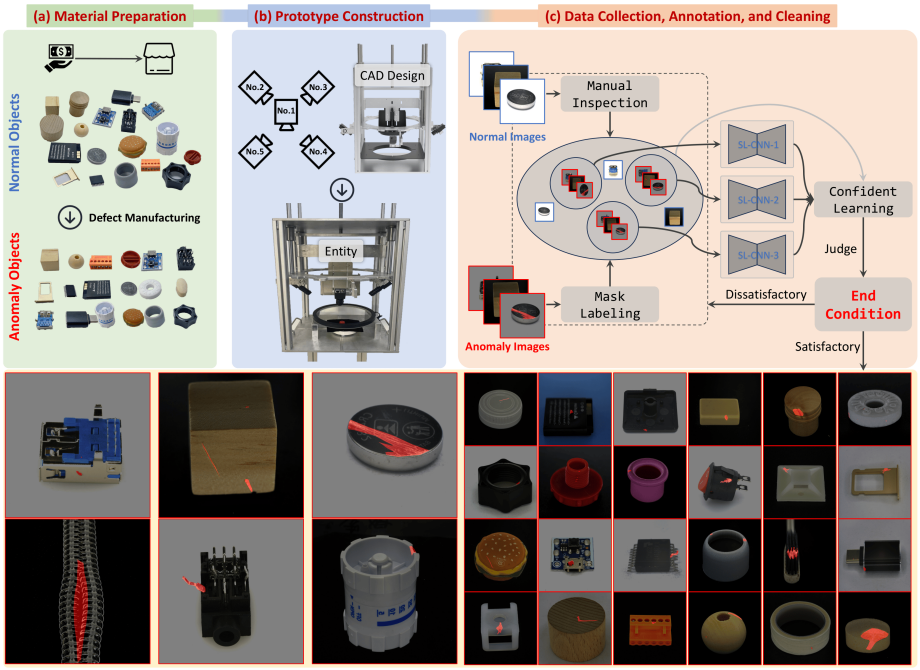

Figure 4.8: Pipeline di acquisizione di Real-IAD: (a) preparazione dei materiali e produzione controllata dei difetti. (b) Progettazione del prototipo di acquisizione multi-vista, con una camera top-down e quattro camere disposte simmetricamente a 45°. (c) Procedura ciclica di raccolta, annotazione e pulizia dei dati tramite ispezione manuale e *confident learning*. In basso, una selezione del dataset finale: 30 categorie, proporzione dei difetti tra 0.01% e 6.75%, rapporto anomalie/normali tra 1:1 e 1:10. Le regioni anomale sono evidenziate in rosso.

2. Fully Unsupervised IAD (FUIAD). Estensione realistica del setting UIAD dove il training set contiene *rumore* (una frazione sconosciuta di campioni anomali mescolati ai normali), riflettendo scenari dove non è possibile garantire purezza assoluta del training set.

FUIAD è particolarmente rilevante per deployment industriale, dove la certificazione manuale di ogni immagine di training può essere proibitiva. I metodi devono essere robusti a contaminazione parziale del training set.

Metriche. **AUROC** (image-level e pixel-level) e **AUPRO (Area Under PRO curve)**, che integra la metrica PRO su tutti i threshold.

Risultati.

1. PatchCore mantiene performance elevate ($> 95\%$ AUROC)
2. Metodi data-augmentation-based (SimpleNet) mostrano robustezza superiore in FUIAD
3. Multi-view fusion migliora significativamente la detection rispetto a single-view

Impatto. Real-IAD stabilì un nuovo standard di scala e realismo, spingendo la ricerca verso soluzioni che non solo massimizzano le metriche su benchmark controllati, ma mantengono robustezza e affidabilità in condizioni operative realistiche.

4.3.7 MVTec AD 2 (2025)

MVTec Anomaly Detection 2 (MVTec AD 2) [6] rappresenta l'evoluzione del benchmark MVTec AD originale, introducendo sfide che riflettono problemi aperti nella pratica industriale.

Caratteristiche. Il dataset contiene **8,004 immagini RGB ad alta risoluzione** distribuite su 8 scenari:

1. **Bulk goods**: materiali sfusi (*Wall Plugs*, *Walnuts*, *Rice*) con oggetti sovrapposti e occlusi
2. **Textured objects**: oggetti (*Fabric*, *Sheet Metal*) con texture complesse ed elevata variabilità
3. **Reflective metal objects**: superfici metalliche (*Sheet Metal*, *Can*) lucide con riflessi
4. **Transparent objects**: oggetti trasparenti (*Vial*, *Fruit Jelly*) che inducono distorsioni

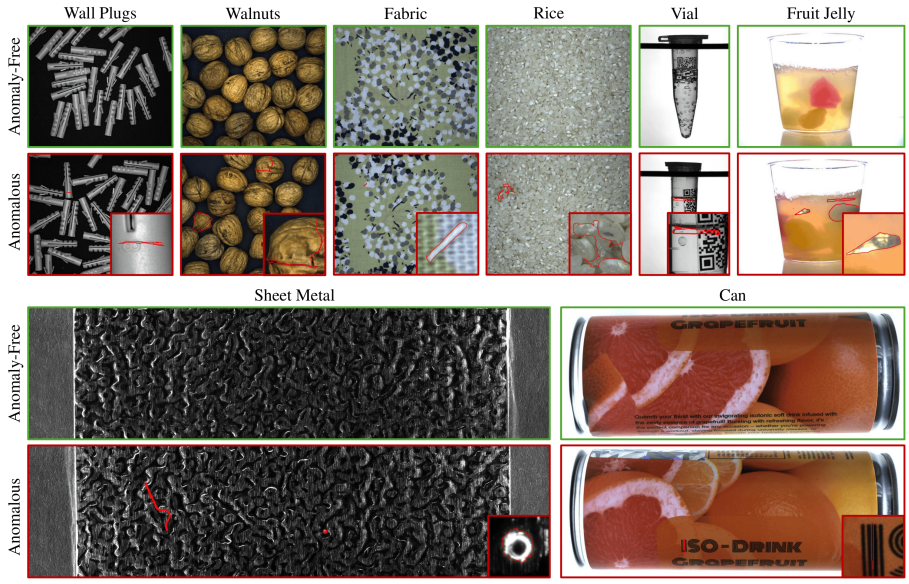

Figure 4.9: Esempi degli oggetti di MVTec AD 2. Per ciascuna categoria sono mostrate un'immagine anomaly-free e un'immagine anomala (bordo rosso), con ingrandimento della regione difettosa e relativa maschera di ground truth pixel-precisa.

Ogni scenario presenta sfide specifiche che causano fallimenti nei metodi standard. I difetti variano per tipologia e includono imperfezioni superficiali come graffi e fori, contaminazione da corpi estranei e parti mancanti.

MVTec AD 2 introduce una serie di innovazioni:

Anomalie ai Bordi dell'Immagine. I difetti appaiono su tutta l'estensione dell'immagine, *inclusi i margini*. Questo penalizza i metodi che si basano su patch o convoluzioni con padding (es. PatchCore), poiché il contesto spaziale risulta incompleto.

Condizioni di Illuminazione Realistiche. Durante l'acquisizione degli oggetti sono impiegati setup comuni nell'ispezione industriale:

- **Back-light illumination** (controluce): illuminazione da dietro l'oggetto (*Vial*, *Fruit Jelly*)
- **Dark-field illumination** (campo scuro): illuminazione angolata che enfatizza difetti superficiali (*Sheet Metal*)

Queste condizioni creano forte variabilità nell'aspetto di oggetti normali, rendendo più difficile distinguere anomalie da variazioni di illuminazione.

Benchmark. MVTec AD 2 valutò metodi state-of-the-art rappresentativi di diverse famiglie:

- **Student-teacher**: EfficientAD [104], Reverse Distillation [102], RD++ [103]
- **Memory-bank**: PatchCore [81]
- **Normalizing Flow**: MSFlow [94]
- **Synthetic anomalies**: SimpleNet [105], Dual Subspace Re-Projection [143]

Nuove Metriche. MVTec AD 2 introdusse metriche più rigorose:

- **AU-PRO (FPR 5%)**: threshold-independent metric che integra PRO fino a un False Positive Rate del 5%, enfatizzando precisione in regimi operativi realistici
- **F1-score**: metrica threshold-dependent calcolata su una soglia fissa, utile per confronti diretti in condizioni operative specifiche.

Risultati Chiave. I risultati rivelarono:

1. Nessun metodo raggiunge 60% AU-PRO (media 52.6%), contro una media di 92.4% AU-PRO sullo stesso set di metodi valutati su MVTec AD
2. Oggetti trasparenti e riflettenti (Vial, Fruit Jelly,Sheet Metal, Can) rappresentano le categorie più difficili, con performance significativamente inferiori alla media
3. Condizioni di illuminazione variabili aumentano significativamente i falsi positivi
4. EfficientAD ottiene le migliori performance complessive (58.7% AU-PRO), unico metodo a superare la soglia del 58%

La Tabella 4.2 confronta le caratteristiche di MVTec AD 2 con altri dataset principali, evidenziando le sue peculiarità.

4.4 Dataset 3D

Sebbene i dataset 2D RGB abbiano inizialmente dominato la ricerca, le immagini a colori presentano un limite intrinseco: le anomalie che si manifestano esclusivamente nella geometria della superficie (ammaccature, deformazioni, graffi profondi) possono risultare invisibili o ambigue in una proiezione 2D. L'integrazione di informazioni tridimensionali (depth map, point cloud, normal map) consente di accedere direttamente alla struttura geometrica dell'oggetto, rendendo rilevabili categorie di difetti altrimenti inaccessibili.

Table 4.2: Confronto qualitativo dei dataset 2D basato su diversi criteri. MVTec AD 2 introduce immagini reali con alta variabilità nello stato anomaly-free e diverse condizioni di illuminazione. Adattato da [6].

	MVTec AD	**MVTec LOCO AD**	**VisA**	**MVTec 3D AD**	**Eye-candies**	**Real-IAD**	**MVTec AD 2**
Real images	✓	✓	✓	✓	×	✓	✓
High Variability*	×	✓	×	✓	✓	×	✓
Defects at Borders	×	×	×	×	×	×	✓
Variation #Objects/image*	×	×	×	×	×	×	✓
Structural Defects	✓	✓	✓	✓	✓	✓	✓
Logical Defects	×	✓	×	×	×	×	×
2D Data	✓	✓	✓	✓	✓	✓	✓
3D Data	×	×	×	✓	✓	×	×
Different Lighting	×	×	×	×	✓	×	✓
Transparent Objects	×	×	×	×	✓	×	✓

*Variazione ammessa nei dati normali utilizzati durante training e validation

4.4.1 MVTec 3D-AD (2021)

MVTec 3D Anomaly Detection (MVTec 3D-AD) [2] estese il paradigma MVTec AD al dominio 3D, introducendo il primo dataset completo per l'anomaly detection non supervisionata su dati tridimensionali.

Caratteristiche. Il dataset contiene **4,147 scansioni 3D ad alta risoluzione** di **10 categorie di oggetti reali**, acquisite con un sensore 3D industriale ad alta risoluzione usando luce strutturata. Le categorie coprono un ampio spettro di caratteristiche: (1) oggetti con variazioni naturali da campione a campione (*bagel*, *carota*, *biscotto*, *pesca*, *patata*), (2) oggetti dall'aspetto standardizzato ma facilmente deformabili (*schiuma*, *corda*, *pneumatico*), (3) oggetti rigidi (*pressacavo*, *tassello*). Ogni campione include:

- **Point cloud 3D**: coordinate (x, y, z) della superficie dell'oggetto, fornite come immagine a tre canali
- **Immagine RGB**: informazioni di texture associate a ciascun punto 3D

Le anomalie si manifestano nella **struttura geometrica** dell'oggetto: graffi (scratches), ammaccature (dents), buchi (holes), contaminazioni (contaminations) e deformazioni strutturali, per un totale di 41 tipologie di difetti. Ground truth annotations (maschere binarie pixel-precise, ottenute proiettando le annotazioni 3D nello spazio immagine) sono fornite per ciascun campione anomalo del test set.

Metodi Valutati. Il paper valuta estensioni 3D di metodi esistenti, tutti basati su rappresentazioni voxel:

- **Voxel f-AnoGAN**: estensione di f-AnoGAN a griglie voxel 3D
- **Voxel Autoencoder**: autoencoder con convoluzioni 3D su rappresentazioni voxel

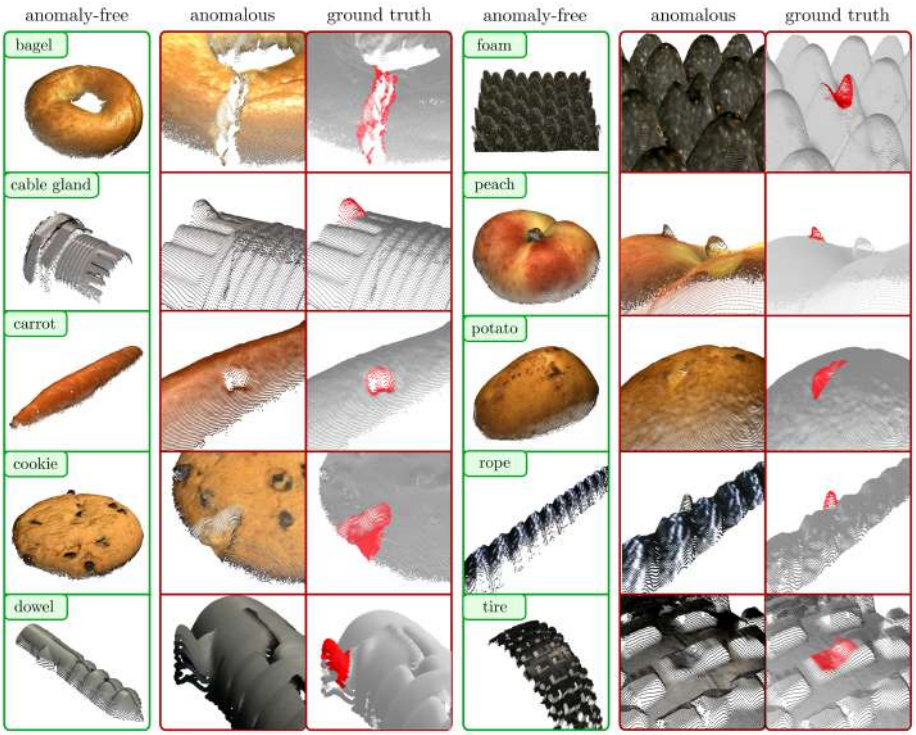

Figure 4.10: Esempi delle 10 categorie di MVTec 3D-AD. Per ciascuna categoria: point cloud anomaly-free con valori RGB proiettati (colonna sinistra), ingrandimento di un campione anomalo del test set (colonna centrale), le regioni anomale evidenziate in rosso (colonna destra). I piani di sfondo sono stati rimossi per maggiore chiarezza visiva.

- **Variation models**: modelli statistici (distanza di Mahalanobis) applicati a dati voxel e depth image

Tutti i metodi furono valutati sia in modalità solo-3D che con l'aggiunta delle informazioni RGB.

Risultati. Gli esperimenti suggerirono che:

1. L'aggiunta di informazioni RGB migliora le performance di *tutti* i metodi rispetto all'uso dei soli dati 3D
2. Difetti puramente geometrici (dents, deformazioni) sono più facilmente rilevabili con point cloud
3. Difetti superficiali (contamination, variazioni cromatiche) richiedono informazioni RGB
4. L'approccio ottimale combina feature 3D e 2D, motivando lo sviluppo di architetture multi-modali

Il benchmark iniziale rivelò inoltre un ampio margine di miglioramento, con performance significativamente inferiori rispetto ai metodi 2D su MVTec AD.

Metriche. AUROC e PRO, calcolati sulle proiezioni 2D delle point cloud. Gli autori raccomandano di integrare la curva PRO fino a un limite massimo di 0.30 ($\text{AU-PRO}_{0.30}$), dato che le regioni anomale sono molto piccole rispetto all'area totale dell'immagine.

Impatto. MVTec 3D-AD dimostrò la complementarità di informazioni 2D e 3D, motivando lo sviluppo di metodi multi-modali come BTF [114] e M3DM [121], che sfruttano entrambe le modalità in modo integrato.

4.4.2 Real3D-AD (2023)

Real3D-AD [116] è il primo dataset composto esclusivamente da **point cloud ad alta precisione**, senza componente RGB. Il suo obiettivo è superare le principali limitazioni di MVTec 3D-AD: precisione insufficiente, presenza di blind spots e assenza di point cloud veri (MVTec 3D-AD fornisce dati RGBD, definiti "2.5D").

Caratteristiche. Il dataset contiene **1,254 campioni** distribuiti su **12 categorie** di oggetti diversi (Airplane, Car, Candybar, Chicken, Diamond, Duck, Fish, Gemstone, Seahorse, , Shell, Starfish, Toffees). Ogni campione è una scansione 3D acquisita con uno scanner binoculare ad alta risoluzione (PMAX-S130), con una precisione di **0.011–0.015 mm** (circa 10 volte superiore a MVTec 3D-AD, 0.11 mm) e copertura a **360° senza blind spots**. Le anomalie sono classificate in due categorie: **incompleteness** e **redundancy**, e annotate punto per punto tramite CloudCompare.

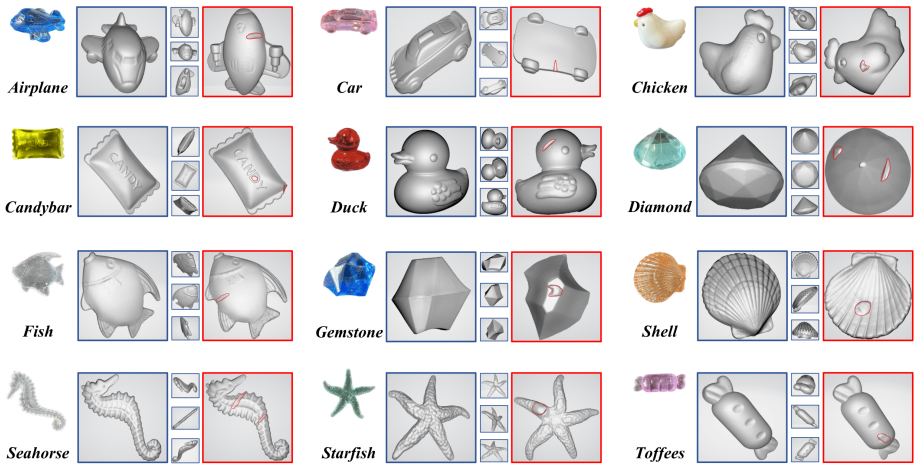

Figure 4.11: Esempi delle 12 categorie di Real3D-AD. Per ciascuna categoria: foto reale dell'oggetto (a sinistra), prototipo del training set con scansioni a 360° (riquadro blu), campione anomalo del test set con anomalia evidenziata in rosso (riquadro rosso).

Scenario Few-Shot. Ogni training set per categoria contiene **4 campioni normali**, costruiti come prototipi dalla fusione di più scansioni a 360°. Questo **few-shot scenario** riflette vincoli reali dell'industria: creare un prototipo ad alta precisione può richiedere fino a due giorni per categoria. I campioni di test, invece, sono acquisiti da un **singolo lato**, simulando le condizioni di una linea produttiva con posizioni di scansione fisse.

ADBENCH-3D. Insieme al dataset, gli autori rilasciarono **ADBENCH-3D**, un benchmark sistematico che definisce il protocollo di valutazione (few-shot training, test su singolo lato) e fornisce un toolkit open-source per preprocessing, metriche e visualizzazione. Il benchmark valutò varianti di BTF, M3DM e PatchCore adattate a point cloud pure, rivelando che la maggior parte non raggiunge il **60% di AUROC object-level**, evidenziando un ampio margine di miglioramento rispetto ai risultati ottenuti su MVTec 3D-AD.

Reg3D-AD. Insieme al dataset, gli autori proposero **Reg3D-AD**, un metodo basato su registrazione di point cloud discusso in Sezione 3.3.2.

Impatto. Real3D-AD stabilì uno standard per la ricerca su point cloud industriali ad alta precisione, distinguendosi nettamente dai dataset RGBD precedenti e motivando lo sviluppo di metodi capaci di elaborare geometria 3D pura senza dipendere da informazioni di colore.

Confronto 3D. La Tabella 4.3 mette a confronto i due dataset, evidenziando come rappresentino approcci complementari: MVTec 3D-AD privilegia la va-

rietà di categorie e la disponibilità di dati RGB, mentre Real3D-AD punta su alta precisione geometrica e copertura completa a 360°.

Table 4.3: Confronto tra i dataset 3D puri per anomaly detection industriale.

	MVTec 3D-AD	**Real3D-AD**
Anno	2021	2023
Modalità	RGB + XYZ (2.5D)	Point cloud puro (3D)
Categorie	10	12
Campioni totali	4,147	1,254
Precisione	0.11 mm	0.011–0.015 mm
Copertura	Singola vista	360° (no blind spots)
Training set	Multi-instance	Few-shot ($\leq$4 prototipi)
Test set	Multi-vista	Singola vista
Annotazioni	Pixel-precise (2D)	Punto per punto (3D)

4.5 Dataset Multimodali

I dataset presentati in questa sezione condividono con MVTec 3D-AD e Real3D-AD l'obiettivo di andare oltre le immagini RGB, ma adottano una strategia diversa: anziché sostituire l'immagine 2D con dati geometrici, la **integrano**. La combinazione di immagini RGB con depth map, normal map, point cloud o viste multiple permette di sfruttare sia le informazioni cromatiche e di texture sia la struttura geometrica dell'oggetto. Questa sezione ripercorre l'evoluzione di questi dataset dal 2022 al 2025, seguendo la progressione da ambienti sintetici controllati (Eyecandies) verso scenari realistici e large-scale (Real-IAD D^3, SiM3D).

4.5.1 Eyecandies (2022)

La raccolta di dataset reali per l'anomaly detection industriale è ostacolata dalla *rarità e imprevedibilità dei difetti*, dai *costi di acquisizione controllata* e dalle *inconsistenze dell'annotazione manuale*. Per questo, Eyecandies [126] introdusse una soluzione alternativa: **dati completamente sintetici** generati proceduralmente, che offrono controllo completo sulle condizioni di acquisizione, annotazioni automatiche e perfettamente precise, e variabilità parametrizzabile sia per le condizioni normali che per le anomalie.

Eyecandies non si propone come sostituto dei dataset reali, né come strumento di pre-training per applicazioni industriali: gli autori forniscono esplicitamente un benchmark per il confronto equo tra metodi, senza avanzare pretese di generalizzazione al dominio reale.

Caratteristiche. Il dataset contiene **15,000 campioni** di oggetti sintetici proceduralmente generati, ciascuno renderizzato in 6 condizioni diverse per un totale di **90,000 rendering foto-realistici**, distribuiti su **10 classi** di dolci (Candy Cane, Chocolate Cookie, Chocolate Praline, Confetto, Gummy Bear, Hazelnut Truffle, Licorice Sandwich, Lollipop, Marshmallow, Peppermint Candy).

La scena simula un **nastro trasportatore industriale** che attraversa una light box. Ogni oggetto è renderizzato con **6 condizioni di illuminazione controllate**:

- una con soli box lights (luci ai quattro angoli della light box),
- quattro con una singola camera light per volta (una per direzione),
- una con tutte le camera lights attive insieme.

Questa struttura è progettata per studiare l'impatto dell'illuminazione direzionale sul rilevamento di anomalie superficiali. Ogni campione, inoltre, include tre modalità:

- **RGB**: immagine renderizzata foto-realistica
- **Depth map**: mappa di profondità (coordinata Z nel riferimento della camera)
- **Normal map**: vettori normali alla superficie, espressi nel riferimento della camera

Tipologie di Anomalie. Le anomalie sono classificate in **3 gruppi**, ciascuno progettato per essere rilevabile principalmente tramite una modalità specifica:

1. **Alterazioni di colore**: macchie e bruciature, visibili nell'immagine RGB
2. **Deformazioni della forma**: come rigonfiamenti o ammaccature, rilevabili nella depth map
3. **Graffi e imperfezioni superficiali**: scalfitture che alterano i vettori normali senza modificare la geometria — visibili solo nelle normal maps

Tutte le anomalie sono annotate con maschere binarie pixel-precise, generate automaticamente dalla pipeline di rendering senza intervento umano.

Esperimenti. Si addestra un deep convolutional autoencoder separatamente per ogni categoria di oggetti, utilizzando diverse combinazioni di input al fine di isolare e verificare il contributo di ciascuna modalità:

- **RGB only**
- **RGB + Depth (RGBD)**

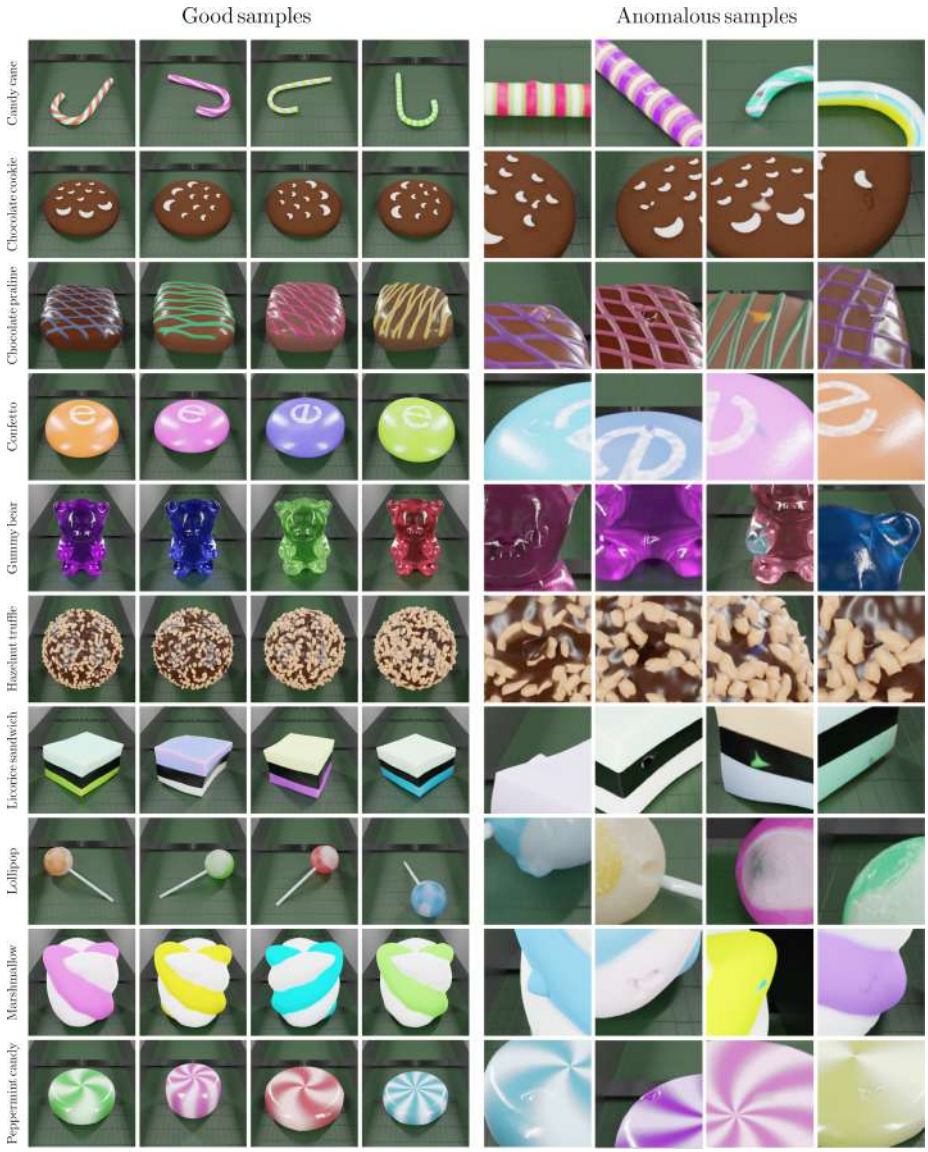

Figure 4.12: Esempi del dataset Eyecandies per ciascuna delle 10 classi. Per ogni categoria, le quattro colonne di sinistra mostrano campioni normali con variabilità intra-classe (colore, forma, orientamento), mentre le quattro colonne di destra mostrano campioni anomali. La variabilità nei campioni normali è ottenuta parametrizzando il pipeline di rendering.

- **RGB + Depth + Normals**

Per ogni categoria, il modello viene addestrato esclusivamente su campioni normali e testato sull'intero test set (contenente sia campioni normali che anomali). L'ipotesi è che le regioni anomale producano un *errore di ricostruzione maggiore*, consentendo di generare mappe di anomalia pixel-wise (distanza L1) e uno score globale per l'intera immagine (calcolato come massimo della mappa di anomalia).

Risultati. I risultati, riportati in termini di image-level e pixel-level AUROC, mostrano che ogni modalità aggiuntiva contribuisce al miglioramento delle performance: RGBD supera RGB-only per le deformazioni di forma, mentre l'aggiunta delle mappe delle normali apporta ulteriori miglioramenti per le imperfezioni superficiali, confermando la corrispondenza progettata tra tipologia di anomalia e modalità di rilevamento.

Impatto e Limitazioni. Eyecandies rimane un riferimento per combinazioni di modalità e per la validazione in condizioni perfettamente controllate. Il *sim-to-real gap* (la differenza tra oggetti sintetici e reali in termini di texture, materiali e rumore di acquisizione) ne limita tuttavia la rappresentatività rispetto agli scenari industriali reali, rendendolo complementare piuttosto che sostitutivo dei benchmark su dati reali.

4.5.2 PAD / MAD (2023)

I dataset e i metodi visti finora assumono che gli oggetti siano sempre nella stessa posizione e orientamento (*pose-aligned assumption*), il che limita la capacità di rilevare anomalie potenzialmente occluse da pose arbitrarie. In scenari reali (robot pick-and-place, nastri trasportatori non allineati, ispezioni manuali) questa assunzione non regge, causando falsi positivi nei metodi convenzionali.

Zhou, Li, Jiang, *et al.* [144] introducono quindi il primo setting e dataset esplicitamente progettati per l'anomaly detection in condizioni di posa arbitraria: il dataset **MAD** (*Multi-pose Anomaly Detection*) e il benchmark **PAD** (*Pose-Agnostic Anomaly Detection*).

Caratteristiche. MAD contiene **oltre 11,000 immagini RGB ad alta risoluzione** di **20 giocattoli LEGO a forma di animale** (tra i quali Gorilla, Unicorn, Mallard, Turtle, Whale, Bird, Owl, Puppy), scelti per la varietà di geometrie e colori. Il dataset è suddiviso in:

- **MAD-Sim**: generato proceduralmente con Blender, con un rapporto tra i campioni di training e test pari circa a 2:1
- **MAD-Real**: dati acquisiti in scene reali, con circa un quinto dei campioni di MAD-Sim per un rapporto tra esempi di training e test di circa 4:1

Figure 4.13: Il dataset MAD. (a) I 20 giocattoli LEGO a forma di animale nelle loro pose simulate; (b) esempi delle tre tipologie di anomalie (burrs, stains e missing parts) su tre categorie distinte, con le regioni anomale evidenziate in rosso.

Tipologie di Anomalie. MAD include tre tipologie di anomalie:

- **Burrs**: sbavature sui bordi
- **Stains**: macchie superficiali
- **Missing parts**: componenti mancanti

Le anomalie sono sia simulate (generate in Blender e modificate manualmente in Photoshop) che reali (difetti autentici su pezzi LEGO provenienti dalla linea produttiva).

OmniposeAD. Il paper propose **OmniposeAD**, che introduce per la prima volta le **Neural Radiance Fields (NeRF)** [145] nel paradigma dell'anomaly detection. NeRF codifica l'oggetto anomaly-free da diverse pose, permettendo di generare immagini di riferimento normali da qualsiasi angolazione (Figura 4.14). Al test, l'immagine query viene confrontata con la ricostruzione NeRF della posa corrispondente per localizzare le anomalie. Questo approccio consente di rilevare anomalie anche da pose mai viste durante il training.

Metodi. PAD valuta metodi delle due famiglie principali, coprendo 8 paradigmi distinti:

- feature embedding-based: PatchCore, STFPM, FastFlow, CFlow, CFA, CutPaste
- reconstruction-based: DRAEM, FAVAE, OCR-GAN, UniAD

Risultati e Metriche. Il benchmark valuta i metodi tramite AUROC image-level e pixel-level nel setting PAD. I risultati mostrano che i metodi standard degradano significativamente rispetto alle loro performance su dataset pose-aligned: metodi basati su pseudo-anomalie come DRAEM e CutPaste sono particolarmente penalizzati, poiché la stessa regione può apparire diversa a seconda

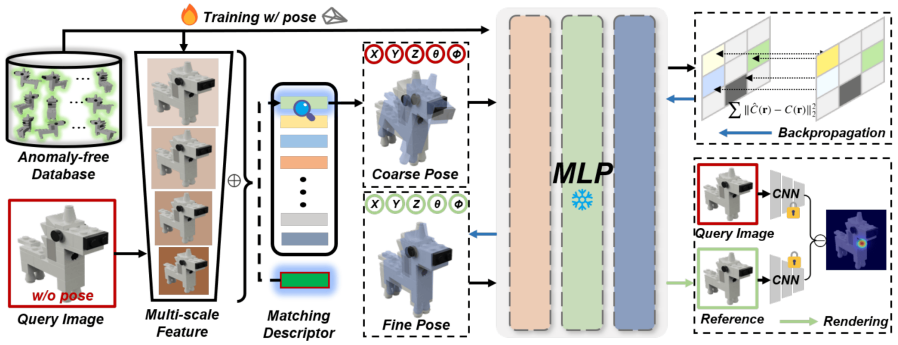

Figure 4.14: Pipeline di OmniposeAD. (1) Un Neural Radiance Field viene addestrato su viste multi-posa dell'oggetto normale. (2) L'immagine query attraversa una stima della posa in due stadi (coarse-to-fine): prima tramite matching con il database anomaly-free, poi raffinata con iNeRF. (3) NeRF renderizza l'immagine di riferimento normale alla posa stimata, che viene confrontata con la query per la localizzazione delle anomalie.

della posa, rendendo difficile caratterizzare le anomalie tramite semplice data augmentation. Per la localizzazione pixel-level, diversi metodi mantengono performance accettabili, mentre la detection image-level rimane una sfida aperta per tutti i metodi considerati. OmniposeAD ottiene le migliori performance su entrambe le metriche, superando CFlow di +7.0/+19.6 e UniAD di +8.7/+28.7 rispettivamente in pixel-level e image-level AUROC.

Impatto e Limitazioni. PAD evidenziò che i metodi standard degradano significativamente sotto variazioni di posa, e che la modellazione esplicita della geometria 3D tramite NeRF è una direzione promettente per l'anomaly detection in scenari non controllati. Il dataset presenta tuttavia alcune limitazioni: include esclusivamente oggetti rigidi standard, senza la variabilità naturale tipica dei prodotti industriali reali, e la complessità nella raccolta di campioni con caratteristiche diverse ne limita la diversità. OmniposeAD fatica a generalizzare a categorie di oggetti non viste durante il training, lasciando aperta la questione della generalizzazione a nuove classi.

4.5.3 RAD (2024)

L'anomaly detection è una capacità necessaria in ambienti industriali, tuttavia la maggior parte dei dataset esistenti sono raccolti in condizioni controllate, con pose fisse e sensori perfettamente calibrati. In scenari reali con robot industriali, le osservazioni sono influenzate da fattori esterni (illuminazione variabile, pose casuali degli oggetti, sfondi inconsistenti) e interni (calibrazione imperfetta della camera, imprecisioni nei movimenti). **RAD** (*Realistic Anomaly Detection*) [146] introduce il primo dataset multi-view per anomaly detection acquisito con

un braccio robotico reale, colmando questo gap e fornendo dati acquisiti in condizioni realisticamente rumorose.

Acquisizione. I dati sono acquisiti con un braccio robotico Franka Emika Panda equipaggiato con una camera Intel RealSense D415 (1280×720). Ogni oggetto è catturato da **68 viewpoints predefiniti** che coprono una rotazione completa a 360°. Rispetto ai multi-camera rig o agli scanner rotanti usati in dataset precedenti (PAD, Real3D-AD), il braccio robotico garantisce copertura multi-view densa con un singolo sensore senza manipolare l'oggetto. Nonostante la camera sia RGB-D, solo le immagini RGB sono rilasciate a causa del rumore eccessivo nelle misure di profondità in condizioni reali. Ogni immagine, infine, è accompagnata dai metadati sulla posizione della camera.

Caratteristiche. Il dataset contiene **5,848 immagini RGB** distribuite su **13 categorie** di oggetti quotidiani, tra cui oggetti da cucina, giocattoli e articoli di consumo (ad esempio, tazza, cucchiaio, cover per telefono, pallina da tennis, etc). Sono presenti **4 tipologie di difetti**:

1. *graffi* e incisioni superficiali
2. *parti mancanti*
3. *macchie* e scolorimento localizzato
4. *deformazioni meccaniche*, escluse le categorie in materiale rigido (binder-clip, box, charger e phonecase)

Le maschere di annotazione pixel-wise sono generate confrontando le immagini difettose con le corrispondenti viste normali.

Metodi. RAD valutò tre famiglie di approcci nel setting pose-agnostic:

- **2D feature-based**: vengono valutati otto metodi non supervisionati operanti su immagini 2D RGB (CFlow, EfficientAD, FastFlow, PaDiM, PatchCore, ReverseDistillation, STFPM, UFlow) e tre varianti zero-shot basate su CLIP (WinCLIP, AdaCLIP, VCPCLIP)
- **3D reconstruction**: SplatPose e PIAD, che costruiscono rappresentazioni 3D tramite 3D Gaussian Splatting
- **VLM pipelines**: Qwen2.5-VL e ChatGPT-4o, valutati con un protocollo a tre step (classificazione delle anomalie a livello di immagine, predizione dei bounding box dell'anomalia tramite prompting, conversione dei bounding box in maschere binarie per la valutazione pixel-wise)

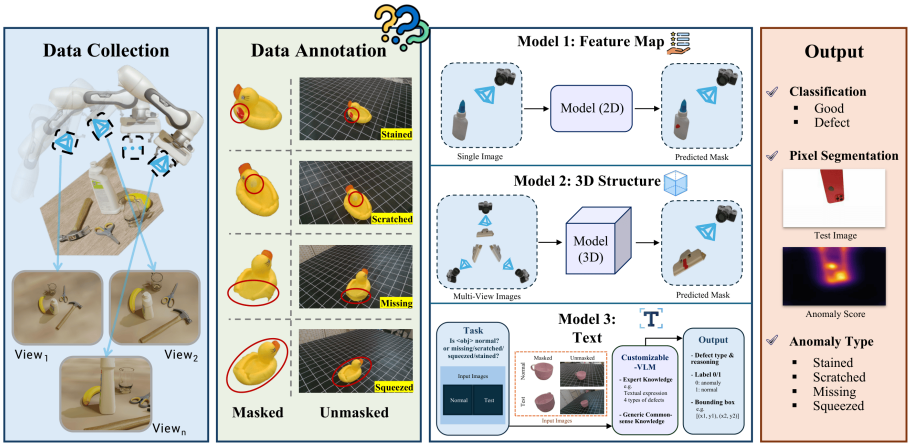

Figure 4.15: Pipeline del benchmark RAD. Il framework integra la raccolta dati multi-view tramite braccio robotico, l'annotazione fine-grained delle anomalie, e tre modelli di modelli: 2D, 3D e VLM testuali. Gli output includono classificazione, segmentazione pixel-wise e identificazione del tipo di anomalia.

Risultati e Metriche. La metrica adottata è AUROC image-level e pixel-level. Il risultato principale è controintuitivo: i metodi **2D feature-embedding** maturi superano significativamente i metodi 3D e VLM a livello image-level, con PatchCore che ottiene il miglior AUROC medio di 0.833. Questo avviene perché i metodi 3D basati su *3D Gaussian Splatting*, pur generando viste di riferimento geometricamente coerenti, rimangono vulnerabili a superfici speculari, simmetrie geometriche e copertura sparsa dei viewpoints: condizioni che producono artefatti di ricostruzione interpretati erroneamente come anomalie. PatchCore, al contrario, cattura implicitamente la variabilità multi-view tramite la sua memory bank di feature normali, risultando più robusto a piccoli disallineamenti. A livello pixel-level il gap si riduce: VCPCLIP raggiunge 0.987 di media, e i metodi 3D (SplatPose 0.977, PIAD 0.984) risultano competitivi, suggerendo che la modellazione geometrica esplicita supporta la localizzazione fine delle anomalie. I VLM rimangono inefficaci su entrambe le metriche (AUROC ≈ 0.52), principalmente per l'assenza di supervisione pixel-level nel pre-training e per l'incapacità di distinguere difetti reali da variazioni di apparenza dovute al cambio di punto di vista. Inoltre, poiché gli attuali VLM non supportano nativamente la generazione di maschere pixel-level, le approssimazioni tramite bounding box introducono ulteriori imprecisioni.

Impatto e Limitazioni. L'analisi individua tre sfide fondamentali per l'ispezione robotica realistica: superfici riflettenti, simmetria geometrica e copertura sparsa dei viewpoints. Le superfici speculari destabilizzano il feature matching e la stima della posa, mentre la simmetria introduce ambiguità nell'ottimizzazione della posa. Questi risultati evidenziano che né l'augmentazione geometrica né

i VLM zero-shot sono sufficienti, motivando approcci futuri che ragionino congiuntamente su apparenza e geometria con consapevolezza dell'incertezza.

4.5.4 Real-IAD D^3 (2025)

Real-IAD D^3 [3] affianca ad immagini RGB ad alta risoluzione e point cloud 3D con precisione micrometrica una terza modalità di rappresentazione dei dati: **pseudo-3D da stereo fotometrico**, con l'obiettivo di catturare difetti superficiali sottili non rilevabili dalle sole modalità 2D o 3D.

Acquisizione. Il sistema integra tre modalità usando un unico setup:

1. **RGB**: camera ad alta risoluzione (3,648×5,472 pixel) per texture e colore
2. **Point cloud 3D**: sistema a luce strutturata DLP a quattro direzioni, con precisione fino a **0.002 mm** e risoluzione massima di 16.2 milioni di punti (5,328×3,040)
3. **Pseudo-3D**: normali alla superficie ottenute tramite stereo fotometrico, calcolando i vettori normali $\mathbf{n}(x, y)$ da quattro immagini acquisite con diverse direzioni di illuminazione:

$$\mathbf{n}(x, y) = (\mathbf{L}^\top \mathbf{L})^{-1} \mathbf{L}^\top \mathbf{I}(x, y)$$

dove $\mathbf{I}(x, y)$ è il vettore delle intensità pixel dalle quattro immagini e $\mathbf{L}$ è la matrice delle direzioni di illuminazione.

Caratteristiche. Il dataset contiene **8,450 campioni**, di cui 5,000 normali e 3,450 anomali, distribuiti su **20 categorie** di componenti industriali reali (che includono, ad esmepio, connettori elettronici, parti meccaniche e sensori) con dimensioni compatte (7–27 mm di lunghezza). Le anomalie sono raggruppate in **69 gruppi di difetti** (come graffi, ammaccature, crepe, parti mancanti e deformazioni), con circa 50 campioni per gruppo e una media di 3.45 tipi di difetti per categoria. La proporzione delle aree con anomalie varia tra 0.46% e 6.39% dell'area totale per quanto riguarda le imamgini 2D, mentre varia tra 0.33% e 7.34% per i punti difettosi nel point cloud.

D^{3}M. Il paper introduce **D^{3}M** (*D^3-Memory*), un framework memory-bank esteso al dominio trimodale, discusso in dettaglio nella Sezione 3.3.4.

Metodi e Risultati. D^{3}M fu confrontato con metodi *single-modality* (CFlow e SimpleNet per RGB, PointMAE per point cloud) e *multi-modality* (AST, PointMAE+PatchCore, M3DM), valutati con **I-AUROC** e **P-AUROC**. I metodi single-modality mostrano limitazioni complementari: il 2D fatica con difetti geometrici come graffi e ammaccature, mentre il 3D manca di dettagli di texture

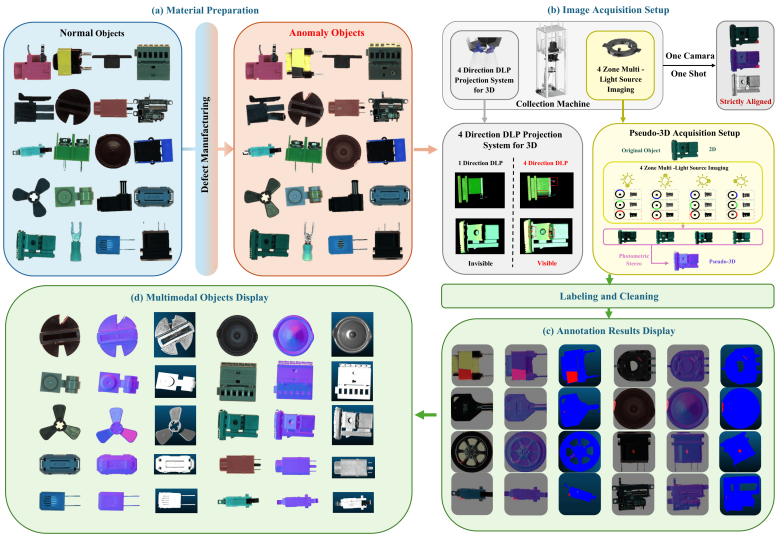

Figure 4.16: Pipeline di acquisizione di Real-IAD D^3. (a) Preparazione dei materiali e creazione dei difetti per le 20 categorie; (b) Setup di acquisizione con singola camera, sistema DLP a quattro direzioni per il point cloud 3D e sorgente multi-luce a quattro zone per il pseudo-3D tramite stereo fotometrico; (c) Annotazione e cleaning con precisione pixel-wise; (d) Visualizzazione multimodale: 2D, pseudo-3D e point cloud 3D.

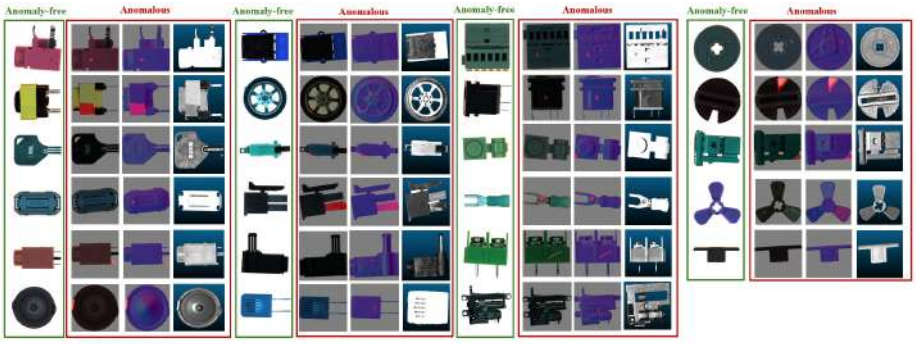

Figure 4.17: Esempi delle 20 categorie del dataset Real-IAD D^3. Per ogni categoria, la prima colonna mostra un campione anomaly-free; le tre colonne successive mostrano campioni anomali nelle tre modalità: RGB, pseudo-3D (stereo fotometrico) e point cloud 3D.

superficiale. La combinazione 2D+3D migliora ma non cattura le irregolarità superficiali sottili.

D^3M raggiunge **0.890 I-AUROC** e **0.937 P-AUROC**, superando M3DM (0.841 / 0.922) e PointMAE+PatchCore (0.812 / 0.905), con la modalità pseudo-3D che fornisce il contributo incrementale decisivo: la sola combinazione 2D+pseudo-3D (PatchCore: 0.839 I-AUROC) supera già il solo 2D (0.747).

Impatto. Real-IAD $\mathbf{D^3}$ stabilisce che i dati pseudo-3D da stereo fotometrico sono una modalità genuinamente complementare a RGB e point cloud, catturando informazioni superficiali non accessibili alle altre due. La piattaforma di acquisizione integrata, che allinea naturalmente le tre modalità senza calibrazione aggiuntiva, rappresenta un contributo metodologico rilevante per la raccolta di dataset multimodali industriali.

4.5.5 SiM3D (2025)

SiM3D (*Single-instance Multiview Multimodal and Multisetup*) [4] introduce il primo benchmark per anomaly detection e segmentation (ADS) che integra simultaneamente informazione **multi-view** e **multimodale**, proponendo sia un dataset ad alta risoluzione che un protocollo di benchmark incentrato su uno scenario specifico, la **single-instance anomaly detection**, in cui un solo oggetto nominale è disponibile per il training.

Acquisizione. Il dataset è acquisito tramite lo scanner 3D industriale *Atos Q* di ZEISS, montato su un braccio robotico ad alta ripetibilità. Il sensore è composto da una coppia stereo di due camere grayscale da **12 Mpx** (4096×3000 pixel) e un proiettore di luce strutturata. I point clouds possono raggiungere fino a 12 milioni di elementi con una distanza inter-punto tra 0.04 e 0.15 mm.

Per ciascuna istanza vengono acquisite da **12 a 36 viste** dell'oggetto, ottenendo per ogni vista:

- un'immagine grayscale I_i,
- una nuvola di punti P_i,
- la posa di acquisizione $[R_i|T_i]$ (la coppia rotazione-traslazione che allinea la vista i alla vista di riferimento v_1)

Il sensore fornisce inoltre un'unica **mesh integrata** M per istanza, ottenuta registrando e fondendo le nuvole di punti di tutte le viste acquisite.

Caratteristiche. Il dataset comprende **333 istanze** appartenenti a **8 tipologie di oggetti** (Plastic Stool, Rubbish Bin, Wicker Vase, Bathroom Furniture, Container, Plastic Vase, Wooden Stool, Sink Cabinet). Le anomalie sono state introdotte *manualmente* simulando difetti realistici su metà delle istanze

- alterazioni dell'**aspetto**: modifiche alla vernice

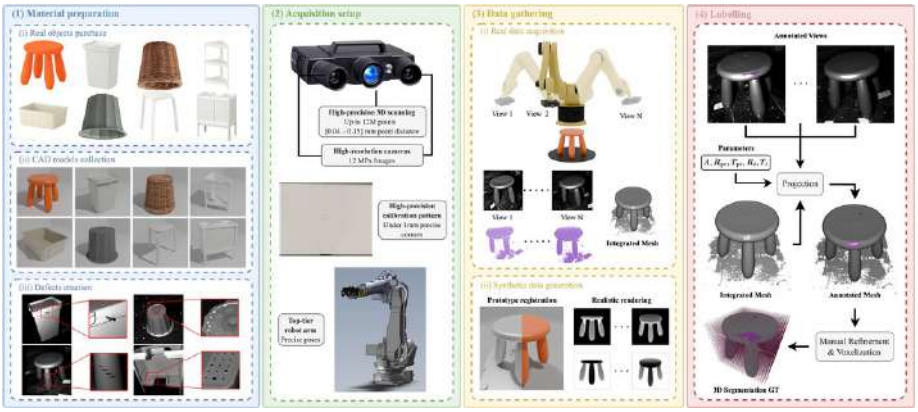

Figure 4.18: Pipeline di creazione del dataset SiM3D. (1) Acquisto degli otto tipi di oggetti, ciascuno associato a un modello CAD, e introduzione manuale di difetti realistici su metà delle istanze; (2) Setup di acquisizione con scanner 3D ad alta precisione e camere ad alta risoluzione montate su braccio robotico; (3) Scansioni a 360° di ciascuna istanza per training e test set reali, e rendering da modello CAD per il training set sintetico; (4) Generazione delle ground truth di segmentazione 3D tramite pipeline mista 2D–3D con raffinamento manuale e voxelizzazione finale.

- alterazioni della **geometria**: ammaccature
- **entrambi**: contaminazioni

Dati sintetici. Per ogni tipologia di oggetto è disponibile un **modello CAD** ufficiale del produttore. Il modello viene allineato a una mesh reale di riferimento e caricato in Blender, dove tramite il renderer Cycles vengono renderizzate immagini dalle stesse pose v_i delle acquisizioni reali. I render RGB sono convertiti in immagini grayscale per coerenza con il sensore reale; vengono renderizzate anche depth map,poi proiettate in nuvole di punti tramite i parametri intrinseci della camera.

Setup sperimentali. SiM3D definisce due configurazioni di training distinte:

- `real2real`: il training set è una singola istanza nominale reale e il test set comprende tutte le altre (nominali e anomale).
- `synth2real`: il training set è il rendering sintetico del modello CAD e il test set è identico al setup `real2real`.

Annotazione e rappresentazione 3D. La ground truth è prodotta tramite una **strategia di annotazione in due fasi**:

- prima si annotano manualmente le viste 2D in cui le anomalie sono visibili

- poi le maschere 2D vengono proiettate sulla mesh integrata tramite i parametri di calibrazione noti
- infine, la mesh viene convertita in una **voxel grid** con risoluzione di **2 mm** per voxel

Il task di output è quindi un **Anomaly Volume** $V \in \mathbb{R}^{X \times Y \times Z}$, anziché la tradizionale Anomaly Map 2D: ogni voxel porta uno score di anomalia, consentendo la localizzazione precisa dei difetti nello spazio tridimensionale.

Metodi e Risultati. Per stabilire baseline sul task multiview 3D ADS, metodi single-view consolidati vengono estesi **proiettando le Anomaly Map 2D nello spazio 3D** e aggregandole nella voxel grid tramite max pooling per voxel.

Dai risultati ottenuti, si può notare che i metodi che processano solo immagini RGB performano meglio rispetto a quelli multimodali: i backbone 3D esistenti sono stati progettati per point cloud a bassa risoluzione (∼8K punti), mentre SiM3D ne fornisce ∼2.3M dopo downsampling, rendendo inutilizzabili i feature extractor nativi di M3DM e CFM. AST è l'unica eccezione, operando su depth map che conservano la struttura 2D.

PatchCore raggiunge il miglior detection (**0.754** I-AUROC in `real2real`), AST la miglior segmentation (**0.699** V-AUPRO@1%). Il gap tra i due setup è marcato per il detection (0.754 vs. 0.540) ma quasi nullo per la segmentation (0.699 vs. 0.695).

Impatto. SiM3D individua tre direzioni di ricerca ancora aperte nel campo dell'ADS 3D

1. la necessità di metodi capaci di processare l'informazione multi-view direttamente in ingresso in maniera più efficiente, anziché aggregare output single-view a posteriori
2. la mancanza di tecniche robuste al domain shift sintetico-reale: nel setup `synth2real` il gap di detection rispetto a `real2real` è consistente, mentre la segmentation risulta molto meno penalizzata
3. l'assenza di backbone 3D scalabili ad alta risoluzione: i metodi multimodali esistenti sono stati progettati per point cloud a bassa risoluzione e non trasferiscono efficacemente a SiM3D

4.6 Metriche di Valutazione

Nell'introduzione presentata nella Sezione 4.2.4 sono state introdotte le principali famiglie di metriche adottate nei benchmark di anomaly detection. In questa sezione viene approfondita ciascuna metrica, discutendone le proprietà formali, i punti di forza, le limitazioni e i contesti applicativi in cui è preferita.

4.6.1 AUROC (Area Under the Receiver Operating Characteristic Curve)

L'AUROC è la metrica più diffusa nei benchmark di anomaly detection industriale per la sua robustezza allo sbilanciamento tra classi normali e anomale. La curva ROC [66] traccia il tasso di veri positivi (*True Positive Rate*, TPR) in funzione del tasso di falsi positivi (*False Positive Rate*, FPR) al variare della soglia di decisione τ:

$$\mathrm{TPR}(\tau) = \frac{TP(\tau)}{TP(\tau) + FN(\tau)}, \qquad \mathrm{FPR}(\tau) = \frac{FP(\tau)}{FP(\tau) + TN(\tau)}.$$

L'area sotto la curva, compresa nell'intervallo $[0, 1]$, misura la capacità di discriminazione del modello indipendentemente dalla soglia scelta. Un valore pari a 0.5 corrisponde a una classificazione casuale; 1.0 indica separazione perfetta tra normali e anomalie.

Nei benchmark industriali l'AUROC è riportata a due livelli distinti:

- **Image-level AUROC (I-AUROC)**: misura la capacità del modello di classificare un'intera immagine come normale o anomala, assegnandole un singolo score (tipicamente il massimo o la media della mappa di anomalia pixel-wise).
- **Pixel-level AUROC (P-AUROC)**: misura la qualità della localizzazione, trattando ogni pixel come un esempio indipendente e confrontando lo score pixel-wise con la ground truth binaria.

Limitazioni. Il P-AUROC presenta un bias sistematico verso le regioni anomale di grandi dimensioni: una macchia estesa contribuisce proporzionalmente di più alla metrica rispetto a un micro-graffio. Questa asimmetria ha motivato lo sviluppo della metrica PRO (Sezione 4.6.2).

4.6.2 PRO e AU-PRO (Per-Region Overlap)

La metrica PRO [66], [79] affronta il bias dimensionale del P-AUROC. Anziché trattare ogni pixel come indipendente, PRO valuta la qualità della localizzazione pesando equamente le regioni di ground truth indipendentemente dalla loro dimensione [79].

Per il calcolo, la mappa di anomalia viene prima binarizzata applicando una soglia: per ciascuna componente connessa nella ground truth, si calcola la percentuale di overlap con la regione anomala binarizzata. Questo valore viene calcolato per un numero di soglie crescenti fino a raggiungere un false-positive rate medio per-pixel del 30% sull'intero dataset; l'area sotto la curva PRO risultante, normalizzata a un valore massimo di 1, costituisce la misura finale di performance.

I diversi dataset adottano limiti di integrazione differenti in funzione delle caratteristiche dei difetti:

- MVTec 3D-AD usa AU-PRO con $\text{FPR}_{\max} = 0.30$, dato che le anomalie occupano spesso una frazione minima dell'area totale
- MVTec AD 2 adotta $\text{FPR}_{\max} = 0.05$, riducendo la tolleranza per i falsi positivi

4.6.3 sPRO (Saturated Per-Region Overlap)

La metrica sPRO, introdotta in MVTec LOCO AD [1], è una generalizzazione di PRO per le anomalie logiche, in cui la ground truth può essere sovradimensionata rispetto all'area strettamente rilevante ai fini della rilevazione. La metrica introduce una soglia di saturazione $s \in [0, 1]$: il contributo di una regione è saturato a 1 non appena l'overlap supera s, evitando di premiare localizzazioni molto estese rispetto a quelle già sufficientemente precise:

$$\text{sPRO}(P) = \frac{1}{m} \sum_{i=1}^{m} \min\left(\frac{|A_i \cap P|}{s_i,\ 1} \right).$$

Per esempio, nel caso di una puntina da disegno assente, è sufficiente che il metodo segmenti un'area delle dimensioni di una puntina all'interno dello scomparto vuoto, senza bisogno di coprire perfettamente tutti i pixel della regione annotata (potenzialmente più ampia). sPRO penalizza le sotto-segmentazioni (overlap insufficiente a raggiungere la soglia s_i), mentre le predizioni eccessive all'interno della regione annotata non abbassano il punteggio grazie alla saturazione. Le sovrasegmentazioni esterne alle regioni annotate sono catturate dal FPR, riportato insieme a sPRO.

4.6.4 AUPR (Area Under the Precision-Recall Curve)

L'AUPR integra la curva Precision-Recall, che riporta la precisione in funzione del recall [147] al variare della soglia. A differenza dell'AUROC, l'AUPR è particolarmente informativa in scenari con sbilanciamento estremo tra classi: quando le anomalie rappresentano una frazione molto ridotta del test set, l'AUROC può risultare elevata anche per classificatori poco precisi, mentre l'AUPR rimane sensibile alla qualità delle predizioni e più informativa. VisA [138] ha adottato l'AUPR come metrica complementare all'AUROC proprio per questa ragione, dato che le immagini anomale rappresentano circa il 10% del totale.

4.6.5 F1-Score

L'F1-score è la media armonica di precisione e recall [147], calcolata dopo aver fissato una soglia ottimale τ (tipicamente quella che massimizza l'F1 sul validation set):

$$F_1 = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}.$$

Essendo una metrica *threshold-dependent*, l'F1 dipende dalla scelta di τ, il che la rende meno adatta ai confronti tra metodi in assenza di un protocollo di

calibrazione condiviso. Tuttavia, è particolarmente utile per valutare le performance in condizioni operative dove la soglia è fissata a priori da requisiti di sistema (ad esempio, massimo 5% di falsi negativi) come in MVTec AD 2 [6], che la adotta come metrica complementare all'AU-PRO.

4.6.6 MAE (Mean Absolute Error)

Il MAE misura l'errore medio assoluto tra la mappa di anomalia predetta $\hat{A}$ e la ground truth G, entrambe normalizzate in $[0, 1]$ [148] su un'immagine di dimensioni $H \times W$:

$$\text{MAE} = \frac{1}{HW} \sum_{i,j} |\hat{A}_{ij} - G_{ij}|.$$

È adottato principalmente in metodi basati su saliency detection o ricostruzione, come nel dataset Magnetic Tile Defects [125], dove la mappa predetta viene confrontata direttamente con la maschera ground truth. Il MAE risulta meno informativo dell'AUROC in contesti industriali dove le regioni anomale occupano una frazione minima dell'immagine, poiché la metrica è dominata dai pixel normali. Offre tuttavia un'interpretazione diretta della qualità della mappa di anomalia come segnale continuo, indipendentemente da una soglia di binarizzazione.

4.6.7 V-AUPRO (Voxel-level AU-PRO)

V-AUPRO è un'estensione della metrica AU-PRO al dominio 3D, introdotta da SiM3D [4]. La metrica PRO voxel-level è definita come:

$$\text{PRO}_t = \frac{1}{N} \sum_{n=1}^{N} \frac{|\Psi_t \cap V_n|}{|V_n|},$$

dove Ψ_t è l'Anomaly Volume binarizzato alla soglia t, N è il numero di difetti nella ground truth e V_n sono i voxel appartenenti all'n-esimo difetto. La curva PRO viene costruita calcolando questo valore per diverse soglie t, stimate a diversi false positive rate: il FPR è calcolato esclusivamente sui voxel non vuoti, escludendo le regioni della griglia 3D prive di punti.

V-AUPRO si ottiene integrando la curva fino a un FPR massimo dell'1%, normalizzando il risultato nell'intervallo $[0, 1]$. La soglia di integrazione dell'1% è più stringente rispetto al 30% adottato da AU-PRO su MVTec AD e al 5% di MVTec AD 2, riflettendo i requisiti delle applicazioni industriali reali per cui SiM3D è stato progettato.

4.6.8 Considerazioni sulla Scelta delle Metriche

La scelta delle metriche valutate influenza direttamente quali aspetti della performance vengono ottimizzati e la direzione di sviluppo degli algoritmi. La Tabella 4.4 riassume le metriche adottate da ciascun benchmark, evidenziando

come l'evoluzione verso scenari più realistici si rifletta in criteri di valutazione più stringenti.

Un rischio è quello di sviluppare modelli che massimizzano una singola metrica senza generalizzare ad altre. Per questo, i benchmark più recenti riportano sistematicamente metriche complementari (I-AUROC, AU-PRO e F1) fornendo una valutazione più completa e meno suscettibile a ottimizzazioni specifiche.

Table 4.4: Metriche di valutazione adottate dai principali dataset per l'anomaly detection industriale. ✓indica che la metrica è valutata nel paper.

Dataset	I-AUROC	P-AUROC	AU-PRO	sPRO	AUPR	F1/F_β	MAE
Magnetic Tile Defects		✓				✓	✓
MVTec AD	✓	✓	✓				
MPDD	✓	✓					
VisA	✓	✓			✓		
MVTec LOCO AD	✓			✓			
Real-IAD	✓	✓	✓				
MVTec AD 2	✓		✓			✓	
MVTec 3D-AD	✓	✓	✓				
Real3D-AD	✓						
Eyecandies	✓	✓					
PAD / MAD	✓	✓					
RAD	✓	✓					
Real-IAD D^3	✓	✓					
SiM3D	✓		✓*				

*SiM3D adotta V-AUPRO@1% (estensione voxel-level di AU-PRO) anziché AU-PRO standard.
Magnetic Tile Defects: F_β con $\beta^2 = 0.3$. MVTec AD 2: F1-score con soglia fissa.

4.7 Analisi Comparativa dei Dataset

I quattordici dataset analizzati in questo capitolo rappresentano l'evoluzione degli ultimi sette anni (2018–2025) che riflette la complessità crescente delle sfide proposte alla ricerca. La Tabella 4.5 ne sintetizza le caratteristiche principali, consentendo un confronto diretto sugli aspetti più rilevanti: modalità di acquisizione, ampiezza del dataset, tipologia di anomalie, caratteristiche distintive e metriche di riferimento.

Osservazioni finali. L'analisi comparativa dei dataset rivela l'evoluzione di tre caratteristiche principali

1. **Aumento della complessità dei dati**: dai 1.344 campioni in scala di grigi di Magnetic Tile alle 150.000 immagini RGB multi-view di Real-IAD, fino alle 333 istanze 3D complete di SiM3D. Questa progressione sottolinea l'ampliamento e la specificazione maggiore dei dati considerati

2. **Aumento delle anomalie**: i dataset di prima generazione si concentrano sui difetti superficiali (graffi, crepe, macchie), facilmente annotabili ed evidenti. I benchmark successivi introducono progressivamente anomalie logiche (MVTec LOCO AD), difetti geometrici sempre più sottili (Real3D-AD), imperfezioni superficiali rilevabili con stereo fotometrico (Real-IAD

Table 4.5: Riepilogo comparativo dei principali dataset per IAD (2018–2025).

Anno	Dataset	Modalità	Cat.	Campioni	Tipo anomalie	Caratteristiche principali
			Dataset 2D RGB			
2018	Magnetic Tile	2D grayscale	6	1,344	Superficie (crepe, fori, sbavature)	Primo dataset AD industriale dedicato; introduce MCuePushU; metriche: AUC, MAE, F_β.
2019	**MVTec AD**	2D RGB	15	5,354	Superficie + strutturale (73 tipologie)	**Benchmark universale**; annotazioni pixel-precise; separa training/test; introduce AUROC e PRO.
2021	MPDD	2D RGB	6	1,346	Graffi, buchi, ruggine, parti mancanti	Focus su parti metalliche; gap di -15% AUROC rispetto a MVTec AD; evidenzia limiti di generalizzazione.
2022	VisA	2D RGB	12	10,821	Superficie + strutturale (multi-istanza)	Strutture complesse; protocolli few-shot (1-shot, 5-shot); introduce AUPR e SPD self-supervised.
2022	MVTec LOCO AD	2D RGB	5	3,644	Strutturale + logico	Primo dataset con anomalie logiche; introduce sPRO; richiede reasoning contestuale.
2024	Real-IAD	2D RGB	30	150,000	Multi-tipo (materiali eterogenei)	Large-scale; acquisizione multi-vista (5 angolazioni); introduce setting FUIAD con training rumoroso.
2025	MVTec AD 2	2D RGB	8	8,004	Superficie + bordi immagine	Oggetti trasparenti/riflettenti; illuminazione variabile; anomalie ai margini; AU-PRO@5%.
			Dataset 3D			
2021	MVTec 3D-AD	RGB + XYZ	10	4,147	Geometriche (ammaccature, deformazioni)	Primo dataset 3D industriale; RGB + point cloud; motiva architetture multi-modali.
2023	Real3D-AD	Point cloud	12	1,254	Incompleteness, redundancy	Precisione 0.011 mm; copertura 360°; setting few-shot (4 prototipi); introduce ADBENCH-3D.
			Dataset Multimodali			
2022	Eyecandies	RGB + D + N	10	90,000	Colore, deformazione, imperfezione	Sintetico; controllo totale su illuminazione; 3 modalità (RGB, depth, normal map).
2023	PAD / MAD	2D RGB	20	11,000+	Sbavature, macchie, parti mancanti	Pose-agnostic; primo uso di NeRF per AD; versioni reale (MAD-Real) e sintetica (MAD-Sim).
2024	RAD	2D RGB	13	5,848	Graffi, macchie, parti mancanti, deformazioni	68 viewpoint con braccio robotico; noisy-pose; valuta VLM e 3D Gaussian Splatting.
2025	Real-IAD D^3	RGB + PC + N	20	8,450	69 tipi (graffi, ammaccature, crepe)	Tri-modale; precisione 0.002 mm; stereo fotometrico per pseudo-3D; introduce D^3M.
2025	SiM3D	Gray + PC	8	333 ist.	Aspetto, geometria, contaminazioni	Single-instance training; valutazione voxel-level; setup real2real e synth2real; 12–36 viste.

D = depth map; N = normal map; PC = point cloud; ist. = istanze complete multi-vista.

D^3) e deformazioni di volume (SiM3D). Aumenta quindi la complessità e l'astrazione da gestire per l'architettura.

3. **Protocolli di valutazione più specifici**: dallo scenario one-class standard (training solo su esempi normali) verso setting più articolati: few-shot (Real3D-AD, VisA), noisy training (Real-IAD FUIAD), pose-agnostic (PAD, RAD) e single-instance (SiM3D). Ciascuna di queste varianti affronta un vincolo reale della produzione industriale, dimostrando la necessità di nuovi riferimenti per misurare il progresso del campo.

4.8 Tendenze e Sfide Future

In conclusione, emergono le principali direzioni di ricerca per il futuro del settore, che riflettono la sempre maggiore ricerca di generalizzazione e la necessità di deployment in scenari reali.

Da detection a comprensione. I primi benchmark si concentravano principalmente sulla capacità di separare campioni normali da anomali e di localizzare le anomalie nell'immagine. I dataset più recenti, in particolare MVTec LOCO AD e Real-IAD, ampliano la visione introducendo la comprensione della struttura logica dell'oggetto, delle relazioni tra componenti, dei vincoli di assemblaggio. Questa direzione può portare verso sistemi capaci non solo di rilevare un'anomalia, ma di classificarla, spiegarla e suggerirne la causa, avvicinando l'anomaly detection ad una ispezione interpretabile.

Integrazione multimodale. L'integrazione di modalità complementari (RGB, depth, point cloud, normal map) sta diventando la norma nei benchmark più recenti (MVTec 3D-AD, Eyecandies, Real-IAD D^3, SiM3D). La sfida principale è sviluppare architetture capaci di fondere le varie modalità in modo adattivo, considerando tipo di anomalia e contesto applicativo. I risultati di SiM3D mostrano che i backbone 3D esistenti, progettati per point cloud a bassa risoluzione, non scalano ai dataset ad alta risoluzione: questo apre un ulteriore spazio di ricerca sulle architetture per processare point cloud ad alta risoluzione.

Riduzione del gap sintetico-reale. Eyecandies e MAD-Sim dimostrano i vantaggi di dati sintetici (controllo completo, annotazioni automatiche, scala illimitata), ma anche i suoi limiti: il gap in texture, materiali e rumore di acquisizione riduce la trasferibilità dei modelli. Emergono quindi dataset ibridi come SiM3D, che affiancano rendering da modello CAD e acquisizioni reali, e tecniche di domain adaptation che usano i dati sintetici come fonte di pre-training per poi adattarsi al reale con pochi campioni. Lo sviluppo di modelli generativi può portare verso la generazione di difetti sintetici sempre più realistici, consentendo di aumentare artificialmente i dataset di training.

Setting few-shot e zero-shot. La raccolta di campioni normali ad alta qualità può essere complessa in contesti industriali e la disponibilità di esempi anomali è limitata dalla rarità dei difetti reali. Per questo, i dataset più recenti introducono protocolli few-shot (VisA, Real3D-AD) e single-instance (SiM3D), mentre i vision-language model come WinCLIP e AnomalyCLIP definiscono prospettive zero-shot, con modelli guidati solo da descrizioni testuali. I risultati di RAD mostrano tuttavia che i VLM attuali non competono con i metodi feature-based, lasciando aperta la questione di come integrare ragionamento semantico con feature locali di basso livello.

Massimo realismo. MVTec AD 2 e RAD introducono sfide sempre più complesse: illuminazione variabile, oggetti trasparenti e riflettenti, pose non fisse, bracci robotici. Un benchmark utile non misura le performance in condizioni ideali, ma in quelle in cui i sistemi effettivamente operano: questo porta verso dati acquisiti in ambienti industriali reali, con minore controllo delle condizioni, rispetto a condizioni perfettamente controllate in un laboratorio.

Modelli unificati e generalizzazione. La maggior parte dei metodi analizzati nel Capitolo 2 adotta un approccio per categoria: viene addestrato un modello per ogni tipo di oggetto, che impara il concetto di normalità dagli esempi di quella categoria. Benchmark come Real-IAD (30 categorie) e Real-IAD D^3 (20 categorie) si indirizzano, invece, verso modelli unificati che possono operare su categorie eterogenee senza pre-addestramento. I modelli dovranno avere una rappresentazione più flessibile della normalità, per potersi adattare a texture, materiali e forme molto diverse tra loro.

In sintesi, il panorama dei dataset per l'anomaly detection industriale riflette un campo in crescita, in cui le sfide si spostano dalla detection di difetti superficiali in condizioni controllate verso la comprensione di anomalie complesse in scenari realistici, per rispondere alle esigenze dell'analisi industriale moderna.

Chapter 5

Conclusione

La tesi ha tracciato l'evoluzione del visual anomaly detection industriale attraverso lo sviluppo dei metodi e la progressione dei dataset di riferimento. Il percorso, che parte dal 2018 fino ai lavori più recenti del 2025, evidenzia come il campo sia in continua crescita.

Il **Capitolo 1** ha definito le basi concettuali necessarie a comprendere questa evoluzione: i principi del machine learning, l'architettura delle reti neurali convoluzionali e il ruolo del transfer learning, alla base dei backbone pre-addestrati su ImageNet utilizzati per estrarre rappresentazioni senza supervisione esplicita sulle anomalie.

Il **Capitolo 2** ha seguito lo spostamento progressivo da approcci generativi di prima generazione (autoencoder e GAN, limitati dalla difficoltà di modellare distribuzioni complesse e dall'instabilità dell'addestramento) verso metodi feature-based che usano backbone pre-addestrati (PaDiM, SPADE, PatchCore) con performance superiori e maggiore stabilità. In fasi successive, i normalizing flow hanno introdotto densità esatte sulla distribuzione delle feature normali; le tecniche di sintesi di anomalie (CutPaste, DRAEM) hanno trasformato il problema in una classificazione binaria; i modelli student-teacher avanzati (EfficientAD) hanno bilanciato accuratezza ed efficienza computazionale. Infine, i vision-language model e i framework multimodali per dati 3D hanno aperto nuovi ambiti di ricerca, pur rivelando limiti ancora significativi: i VLM attuali non competono con i metodi feature-based su benchmark standard, e i backbone 3D esistenti non scalano ai dataset ad alta risoluzione.

Il **Capitolo 3** ha seguito l'evoluzione dei dataset. MVTec AD ha definito il primo benchmark di riferimento, portando ad un primo confronto. I dataset successivi hanno progressivamente ampliato il problema: dalle anomalie logiche di MVTec LOCO AD alle informazioni geometriche di MVTec 3D-AD e Real3D-AD, fino agli scenari large-scale e multi-view di Real-IAD e alle condizioni più realistiche di MVTec AD 2 e RAD. Ogni nuovo benchmark ha evidenziato i limiti dei precedenti, orientando la ricerca verso soluzioni più robuste e generalizzabili.

Dall'analisi complessiva emergono alcune osservazioni trasversali. In primo luogo, il rapporto tra le *performance su benchmark controllati* e la *robustezza in scenari reali*: l'ottimizzazione su un singolo dataset non garantisce la generalizzazione. In secondo luogo, l'integrazione multimodale (RGB, depth, point cloud, mappe delle normali) rappresenta la direzione migliore per affrontare sempre maggiori categorie di difetti (che possono essere visibili in alcune modalità e non in altre), sviluppando architetture capaci di fondere le informazioni. In terzo luogo, la disponibilità di dati rimane uno dei vincoli principali: la rarità dei difetti reali e i costi di acquisizione spingono verso approcci few-shot, zero-shot e l'uso di dati sintetici, ma il gap tra simulazione e realtà resta ampio.

Le direzioni di ricerca più rilevanti per il futuro convergono verso tre obiettivi: lo sviluppo di modelli capaci di operare su categorie eterogenee senza readdestramento, la costruzione di architetture 3D scalabili ad alta risoluzione e il passaggio da sistemi di *detection* a sistemi di *comprensione*, capaci di segnalare un'anomalia ma anche di classificarla. Il progresso dipenderà dalla disponibilità di benchmark sempre più rappresentativi delle condizioni operative reali.

In sintesi, il visual anomaly detection industriale si configura oggi come un campo ben definito nella sua metodologia di base, ma ancora aperto sulle sfide più avanzate. Il percorso seguito in questa tesi, dalla detection di graffi su superfici strutturate alla comprensione di assemblaggi complessi in ambienti non controllati, riflette la relazione tra la solidità dei risultati ottenuti e l'ambizione verso sistemi sempre più vicini alle esigenze dell'industria reale.

Bibliography

[1] P. Bergmann, K. Batzner, M. Fauser, D. Sattlegger, and C. Steger, "Beyond dents and scratches: Logical constraints in unsupervised anomaly detection and localization," *International Journal of Computer Vision*, vol. 130, no. 4, pp. 947–969, 2022.

[2] P. Bergmann, X. Jin, D. Sattlegger, and C. Steger, "MVTec 3D-AD: A new benchmark for 3d industrial anomaly detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021, pp. 1094–1104.

[3] W. Zhu, L. Wang, Z. Zhou, *et al.*, "Real-IAD D^3: A real-world 2D/pseudo-3D/3D dataset for industrial anomaly detection," *arXiv preprint arXiv:2504.14221*, 2025.

[4] A. Costanzino, P. Zama Ramirez, L. Lella, *et al.*, "SiM3D: Single-instance Multiview Multimodal and Multisetup 3D Anomaly Detection Benchmark," 2025. arXiv: `2506.21549 [cs.CV]`. [Online]. Available: `https://arxiv.org/abs/2506.21549`.

[5] C. Wang, W. Zhu, B.-B. Gao, *et al.*, "Real-IAD: A real-world multi-view dataset for benchmarking versatile industrial anomaly detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

[6] P. Bergmann, K. Batzner, M. Fauser, D. Sattlegger, and C. Steger, "MVTec AD 2: A challenging anomaly detection dataset," *arXiv preprint*, 2025, Dataset with 8004 images across 8 challenging scenarios including transparent objects, reflective metal objects, bulk goods, and different lighting conditions.

[7] F. Chollet, *Deep Learning with Python*, 2nd. Shelter Island, NY: Manning Publications, 2021, ISBN: 978-1-617-29686-4.

[8] T. M. Mitchell, *Machine Learning*. New York: McGraw-Hill, 1997.

[9] C. M. Bishop, *Pattern Recognition and Machine Learning*. Springer, 2006.

[10] K. P. Murphy, *Machine Learning: A Probabilistic Perspective*. Cambridge, MA: MIT Press, 2012, ISBN: 978-0-262-01802-9.

[11] T. D. Buskirk, A. Kirchner, A. Eck, and C. S. Signorino, "An introduction to machine learning methods for survey researchers," *Survey Practice*, vol. 11, no. 1, 2018. DOI: 10.29115/SP-2018-0001.

[12] T. Hastie, R. Tibshirani, and J. Friedman, *The elements of statistical learning: data mining, inference, and prediction*. Springer, 2009.

[13] K. Lo, F. Hahne, R. R. Brinkman, and R. Gottardo, "Automated gating of flow cytometry data via robust model-based clustering," *Cytometry Part A*, 2009.

[14] P. Berkhin, "A survey of clustering data mining techniques," in *Grouping multidimensional data*, Springer, 2006.

[15] P. Cheeseman, M. Self, J. Kelly, W. Taylor, D. Freeman, and J. Stutz, "Autoclass: A bayesian classification system," in *International Conference on Machine Learning*, 1988.

[16] I. T. Jolliffe, *Principal component analysis*. Springer, 2002.

[17] L. Van der Maaten and G. Hinton, "Visualizing data using t-sne," *Journal of machine learning research*, 2008.

[18] L. McInnes, J. Healy, and J. Melville, "Umap: Uniform manifold approximation and projection for dimension reduction," *arXiv preprint arXiv:1802.03426*, 2018.

[19] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016.

[20] L. Van der Maaten, E. Postma, and J. Van den Herik, "Dimensionality reduction: A comparative review," *Journal of Machine Learning Research*, 2009.

[21] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent Dirichlet allocation," *Journal of Machine Learning Research*, vol. 3, pp. 993–1022, 2003.

[22] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[23] K. Arulkumaran, M. P. Deisenroth, M. Brundage, and A. A. Bharath, "Deep reinforcement learning: A brief survey," *IEEE Signal Processing Magazine*, vol. 34, no. 6, pp. 26–38, 2017.

[24] Y. Li, "Deep reinforcement learning: An overview," *arXiv preprint arXiv:1701.07274*, 2017. [Online]. Available: https://arxiv.org/abs/1701.07274.

[25] V. Mnih, K. Kavukcuoglu, D. Silver, *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015. DOI: 10.1038/nature14236.

[26] S. Levine, C. Finn, T. Darrell, and P. Abbeel, "End-to-end training of deep visuomotor policies," *Journal of Machine Learning Research*, vol. 17, no. 39, pp. 1–40, 2016.

[27] D. Silver, A. Huang, C. J. Maddison, *et al.*, "Mastering the game of go with deep neural networks and tree search," *Nature*, vol. 529, no. 7587, pp. 484–489, 2016. DOI: 10.1038/nature16961.

[28] B. R. Kiran, I. Sobh, V. Talpaert, *et al.*, "Deep reinforcement learning for autonomous driving: A survey," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 6, pp. 4909–4926, 2022.

[29] H. Wang, N. Liu, Y. Zhang, *et al.*, "Deep reinforcement learning: A survey," *Frontiers of Information Technology & Electronic Engineering*, vol. 21, no. 12, pp. 1726–1744, 2020.

[30] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, 2015.

[31] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 8, 2013.

[32] P. Domingos, "A few useful things to know about machine learning," *Communications of the ACM*, vol. 55, no. 10, pp. 78–87, 2012. DOI: 10.1145/2347736.2347755.

[33] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004. DOI: 10.1023/B:VISI.0000029664.99615.94.

[34] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, IEEE, vol. 1, 2005, pp. 886–893. DOI: 10.1109/CVPR.2005.177.

[35] J. G. Daugman, "Uncertainty relation for resolution in space, spatial frequency, and orientation optimized by two-dimensional visual cortical filters," *Journal of the Optical Society of America A*, vol. 2, no. 7, pp. 1160–1169, 1985. DOI: 10.1364/JOSAA.2.001160.

[36] S. Haykin, *Neural Networks and Learning Machines*, 3rd. Upper Saddle River, NJ: Pearson Education, 2009, ISBN: 978-0-13-147139-9.

[37] F. Rosenblatt, "The perceptron: A probabilistic model for information storage and organization in the brain," *Psychological Review*, vol. 65, no. 6, pp. 386–408, 1958. DOI: 10.1037/h0042519.

[38] M. Minsky and S. Papert, *Perceptrons: An Introduction to Computational Geometry*. Cambridge, MA: MIT Press, 1969.

[39] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, 1986. DOI: 10.1038/323533a0.

[40] S. Hochreiter, "Untersuchungen zu dynamischen neuronalen netzen," *Diploma thesis, Institut für Informatik, Technische Universität München*, 1991, L'analisi originale del problema, in tedesco. Il lavoro è poi evoluto nelle LSTM.

[41] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE transactions on neural networks*, vol. 5, no. 2, pp. 157–166, 1994.

[42] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics (AISTATS)*, JMLR Workshop and Conference Proceedings, 2010, pp. 249–256.

[43] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proceedings of the 27th International Conference on Machine Learning (ICML)*, 2010, pp. 807–814.

[44] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics (AISTATS)*, JMLR Workshop and Conference Proceedings, 2011, pp. 315–323.

[45] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *ICCV*, 2015, pp. 1026–1034.

[46] A. L. Maas, A. Y. Hannun, and A. Y. Ng, "Rectifier nonlinearities improve neural network acoustic models," *Proc. icml*, vol. 30, no. 1, p. 3, 2013.

[47] P.-H. Kuo, S.-T. Lin, and J. Hu, "DNAE-GAN: Noise-free acoustic signal generator by integrating autoencoder and generative adversarial network," *International Journal of Distributed Sensor Networks*, vol. 16, no. 5, p. 1 550 147 720 923 529, May 2020, Licensed under CC BY 4.0. DOI: 10.1177/1550147720923529.

[48] J. L. Elman, "Finding structure in time," *Cognitive science*, vol. 14, no. 2, pp. 179–211, 1990.

[49] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997. DOI: 10.1162/neco.1997.9.8.1735.

[50] K. Cho, B. van Merriënboer, C. Gulcehre, *et al.*, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2014, pp. 1724–1734. DOI: 10.3115/v1/D14-1179.

[51] L. Bottou, "Large-scale machine learning with stochastic gradient descent," in *Proceedings of COMPSTAT'2010*, Springer, 2010, pp. 177–186. DOI: 10.1007/978-3-7908-2604-3_16.

[52] B. T. Polyak, "Some methods of speeding up the convergence of iteration methods," *USSR Computational Mathematics and Mathematical Physics*, vol. 4, no. 5, pp. 1–17, 1964.

[53] I. Sutskever, J. Martens, G. Dahl, and G. Hinton, "On the importance of initialization and momentum in deep learning," pp. 1139–1147, 2013.

[54] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[55] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *The journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[56] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International conference on machine learning*, PMLR, 2015, pp. 448–456.

[57] L. Prechelt, "Early stopping—but when?," pp. 55–69, 1998.

[58] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[59] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization," *Journal of machine learning research*, vol. 13, no. 2, 2012.

[60] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998. DOI: 10.1109/5.726791.

[61] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 25, 2012, pp. 1097–1105.

[62] O. Russakovsky, J. Deng, H. Su, *et al.*, "ImageNet large scale visual recognition challenge," *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015. DOI: 10.1007/s11263-015-0816-y.

[63] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014, Published at ICLR 2015.

[64] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 10, pp. 1345–1359, 2010. DOI: 10.1109/TKDE.2009.191.

[65] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks?" In *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 27, 2014, pp. 3320–3328.

[66] P. Bergmann, M. Fauser, D. Sattlegger, and C. Steger, "MVTec AD – a comprehensive real-world dataset for unsupervised anomaly detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 9592–9600. DOI: 10.1109/CVPR.2019.00982.

[67] Z. Li, Y. Yan, X. Wang, Y. Ge, and L. Meng, "A survey of deep learning for industrial visual anomaly detection," *Artificial Intelligence Review*, 2025.

[68] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM computing surveys*, vol. 41, no. 3, pp. 1–58, 2009.

[69] V. Barnett and T. Lewis, *Outliers in Statistical Data*, 3rd. Chichester, UK: John Wiley & Sons, 1994.

[70] L. Ruff, J. R. Kauffmann, R. A. Vandermeulen, *et al.*, "A unifying review of deep and shallow anomaly detection," *Proceedings of the IEEE*, 2021.

[71] R. K. Malaiya, D. Kwon, J. Kim, S. C. Suh, H. Kim, and I. Kim, "An empirical evaluation of deep learning for network anomaly detection," in *Proceedings of the International Conference on Computing, Networking and Communications*, 2018.

[72] W. D. Fisher, T. K. Camp, and V. V. Krzhizhanovskaya, "Anomaly detection in earth dam and levee passive seismic data using support vector machines and automatic feature selection," *Journal of Computational Science*, vol. 20, 2017.

[73] M. Ahmed, A. N. Mahmood, and M. R. Islam, "A survey of anomaly detection techniques in financial domain," *Future Generation Computer Systems*, vol. 55, 2016.

[74] C. Baur, B. Wiestler, S. Albarqouni, and N. Navab, "Fusing unsupervised and supervised deep learning for white matter lesion segmentation," in *Medical Imaging with Deep Learning*, 2019.

[75] S. Min, B. Lee, and S. Yoon, "Deep learning in bioinformatics," *Briefings in Bioinformatics*, vol. 18, no. 5, 2017.

[76] M. A. Pimentel, D. A. Clifton, L. Clifton, and L. Tarassenko, "A review of novelty detection," *Signal Processing*, vol. 99, pp. 215–249, 2014.

[77] D. M. Tax and R. P. Duin, "Support vector data description," *Machine Learning*, vol. 54, no. 1, pp. 45–66, 2004.

[78] H. He, Y. Bai, E. A. Garcia, and S. Li, "ADASYN: Adaptive synthetic sampling approach for imbalanced learning," in *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*, IEEE, 2008, pp. 1322–1328.

[79] P. Bergmann, M. Fauser, D. Sattlegger, and C. Steger, "Uninformed students: Student-teacher anomaly detection with discriminative latent embeddings," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 4183–4192.

[80] R. Chalapathy and S. Chawla, "Deep learning for anomaly detection: A survey," *arXiv preprint arXiv:1901.03407*, 2019, Preprint.

[81] K. Roth, L. Pemula, J. Zepeda, B. Schölkopf, T. Brox, and P. Gehler, "Towards total recall in industrial anomaly detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 14 318–14 328.

[82] T. Defard, A. Setkov, A. Loesch, and R. Audigier, "PaDiM: A patch distribution modeling framework for anomaly detection and localization," in *International Conference on Pattern Recognition*, Springer, 2021, pp. 475–489.

[83] I. Goodfellow, J. Pouget-Abadie, M. Mirza, *et al.*, "Generative adversarial nets," *Advances in Neural Information Processing Systems*, vol. 27, pp. 2672–2680, 2014.

[84] T. Schlegl, P. Seeböck, S. M. Waldstein, U. Schmidt-Erfurth, and G. Langs, "Unsupervised anomaly detection with generative adversarial networks to guide marker discovery," in *International Conference on Information Processing in Medical Imaging*, Springer, 2017, pp. 146–157.

[85] S. Akcay, A. Atapour-Abarghouei, and T. P. Breckon, "GANomaly: Semi-supervised anomaly detection via adversarial training," in *Asian Conference on Computer Vision*, Springer, 2018, pp. 622–637.

[86] T. Schlegl, P. Seeböck, S. M. Waldstein, G. Langs, and U. Schmidt-Erfurth, "F-AnoGAN: Fast unsupervised anomaly detection with generative adversarial networks," *Medical Image Analysis*, vol. 54, pp. 30–44, 2019.

[87] S. Akcay, A. Atapour-Abarghouei, and T. P. Breckon, "Skip-GANomaly: Skip connected and adversarially trained encoder-decoder anomaly detection," in *2019 International Joint Conference on Neural Networks (IJCNN)*, IEEE, 2019, pp. 1–8.

[88] D. P. Kingma and M. Welling, "Auto-encoding variational Bayes," *arXiv preprint arXiv:1312.6114*, 2014, International Conference on Learning Representations (ICLR) 2014.

[89] D. P. Kingma and M. Welling, "An introduction to variational autoencoders," *Foundations and Trends in Machine Learning*, vol. 12, no. 4, pp. 307–392, 2019.

[90] D. Gong, L. Liu, V. Le, *et al.*, "Memorizing normality to detect anomaly: Memory-augmented deep autoencoder for unsupervised anomaly detection," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 1705–1714.

[91] H. Park, J. Noh, and B. Ham, "Learning memory-guided normality for anomaly detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 14 372–14 381.

[92] N. Cohen and Y. Hoshen, "Sub-image anomaly detection with deep pyramid correspondences," *arXiv preprint arXiv:2005.02357*, 2021.

[93] D. Gudovskiy, S. Ishizaka, and K. Kozuka, "CFLOW-AD: Real-time unsupervised anomaly detection with localization via conditional normalizing flows," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2022, pp. 98–107.

[94] J. Yu, Y. Zheng, X. Wang, *et al.*, "FastFlow: Unsupervised anomaly detection and localization via 2d normalizing flows," in *arXiv preprint arXiv:2111.07677*, 2021.

[95] M. Rudolph, B. Wandt, and B. Rosenhahn, "Fully convolutional cross-scale-flows for image-based defect detection," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2022, pp. 1088–1097.

[96] C.-L. Li, K. Sohn, J. Yoon, and T. Pfister, "CutPaste: Self-supervised learning for anomaly detection and localization," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 9664–9674.

[97] V. Zavrtanik, M. Kristan, and D. Skočaj, "DRAEM–a discriminatively trained reconstruction embedding for surface anomaly detection," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 8330–8339.

[98] M. Yang, P. Wu, and H. Feng, "MemSeg: A semi-supervised method for image surface defect detection using differences and commonalities," in *Engineering Applications of Artificial Intelligence*, vol. 119, Elsevier, 2023, p. 105 835.

[99] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, Springer, 2015, pp. 234–241.

[100] Q. Hou, D. Zhou, and J. Feng, "Coordinate attention for efficient mobile network design," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 13 713–13 722.

[101] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 2980–2988.

[102] H. Deng and X. Li, "Anomaly detection via reverse distillation from one-class embedding," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 9737–9746.

[103] T. D. Tien, A. T. Nguyen, N. H. Tran, *et al.*, "Revisiting reverse distillation for anomaly detection," *arXiv preprint arXiv:2201.10703*, 2023.

[104] K. Batzner, L. Heckler, and R. König, "EfficientAD: Accurate visual anomaly detection at millisecond-level latencies," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2024, pp. 5440–5449.

[105] Z. Liu, Y. Zhou, Y. Xu, and Z. Wang, "SimpleNet: A simple network for image anomaly detection and localization," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 20 402–20 411.

[106] J. Jeong, Y. Zou, T. Kim, D. Zhang, A. Ravichandran, and O. Dabeer, "WinCLIP: Zero-/few-shot anomaly classification and segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 19 606–19 616.

[107] Q. Zhou, G. Pang, Y. Tian, S. He, and J. Chen, "AnomalyCLIP: Object-agnostic prompt learning for zero-shot anomaly detection," *International Conference on Learning Representations*, 2024.

[108] A. Radford, J. W. Kim, C. Hallacy, *et al.*, "Learning transferable visual models from natural language supervision," *International Conference on Machine Learning*, pp. 8748–8763, 2021.

[109] M. Bengs, F. Behrendt, J. Krüger, R. Opfer, and A. Schlaefer, "Three-dimensional deep learning with spatial erasing for unsupervised anomaly segmentation in brain MRI," *International Journal of Computer Assisted Radiology and Surgery*, vol. 16, 2021.

[110] J. Simarro Viana, E. de la Rosa, T. Vande Vyvere, D. Robben, D. M. Sima, and CENTER-TBI Participants and Investigators, "Unsupervised 3D brain anomaly detection," in *Brainlesion: Glioma, Multiple Sclerosis, Stroke and Traumatic Brain Injuries*, Springer International Publishing, 2021, pp. 133–142.

[111] C. R. Qi, H. Su, M. Nießner, A. Dai, M. Yan, and L. J. Guibas, "Volumetric and multi-view cnns for object classification on 3d data," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 5648–5656.

[112] Y. Pang, W. Wang, F. E. Tay, W. Liu, Y. Tian, and L. Yuan, "Masked autoencoders for point cloud self-supervised learning," in *Proceedings of the European Conference on Computer Vision (ECCV)*, Springer, 2022, pp. 604–621.

[113] H. Zhao, L. Jiang, J. Jia, P. Torr, and V. Koltun, "Point transformer," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021, pp. 16 259–16 268.

[114] E. Horwitz and Y. Hoshen, "Back to the feature: Classical 3D features are (almost) all you need for 3D anomaly detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2023, pp. 2967–2975.

[115] R. B. Rusu, N. Blodow, and M. Beetz, "Fast point feature histograms (FPFH) for 3D registration," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2009, pp. 3212–3217. DOI: 10.1109/ROBOT.2009.5152473.

[116] J. Liu, G. Xie, R. Chen, *et al.*, "Real3D-AD: A dataset of point cloud anomaly detection," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 36, 2023.

[117] Y.-M. Chu, C. Liu, T.-I. Hsieh, H.-T. Chen, and T.-L. Liu, "Shape-guided dual-memory learning for 3D anomaly detection," in *Proceedings of the 40th International Conference on Machine Learning (ICML)*, ser. Proceedings of Machine Learning Research, PMLR, 2023.

[118] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "PointNet: Deep learning on point sets for 3D classification and segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 652–660. DOI: 10.1109/CVPR.2017.16.

[119] Y. Lin, Y. Chang, X. Tong, *et al.*, "A survey on RGB, 3D, and multimodal approaches for unsupervised industrial image anomaly detection," *arXiv preprint arXiv:2410.21982*, 2025, Preprint, versione aggiornata marzo 2025.

[120] M. Rudolph, T. Wehrbein, B. Rosenhahn, and B. Wandt, "Asymmetric student-teacher networks for industrial anomaly detection," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2023, pp. 2592–2601. DOI: 10.1109/WACV56688.2023.00262.

[121] Y. Wang, J. Peng, J. Zhang, R. Yi, Y. Wang, and C. Wang, "Multimodal industrial anomaly detection via hybrid fusion," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023, pp. 8032–8041. DOI: 10.1109/CVPR52729.2023.00776.

[122] A. Dosovitskiy, L. Beyer, A. Kolesnikov, *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," in *International Conference on Learning Representations (ICLR)*, 2021.

[123] A. Costanzino, P. Zama Ramirez, G. Lisanti, and L. Di Stefano, "Multimodal industrial anomaly detection by crossmodal feature mapping," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. DOI: 10.1109/CVPR52733.2024.00645.

[124] M. Caron, H. Touvron, I. Misra, *et al.*, "Emerging properties in self-supervised vision transformers," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021, pp. 9650–9660.

[125] Y. Huang, C. Qiu, and K. Yuan, "Surface defect saliency of magnetic tile," in *2018 IEEE 14th International Conference on Automation Science and Engineering (CASE)*, IEEE, 2018, pp. 612–617.

[126] L. Bonfiglioli, M. Toschi, D. Silvestri, N. Fioraio, and D. De Gregorio, "The Eyecandies Dataset for Unsupervised Multimodal Anomaly Detection and Localization," in *Proceedings of the Asian Conference on Computer Vision*, 2022, pp. 3586–3602.

[127] C. Ding, G. Pang, and C. Shen, "Real-IAD: A real-world multi-view dataset for benchmarking versatile industrial anomaly detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024, pp. 20 094–20 104.

[128] J. Liu, X. Cao, Z. Li, C. Liu, G. Wang, and W. Wang, "RAD: A comprehensive real-world multi-view rgb-based dataset for anomaly detection in robotic inspection," *arXiv preprint arXiv:2407.17251*, 2024.

[129] C. Yang, L. Zhang, H. Lu, X. Ruan, and M.-H. Yang, "Saliency detection via graph-based manifold ranking," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3166–3173, 2013.

[130] J. Zhang, S. Sclaroff, Z. Lin, X. Shen, B. Price, and R. Mech, "A minimum barrier salient object detection at 80 FPS," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 1609–1616.

[131] H. Jiang, J. Wang, Z. Yuan, Y. Wu, N. Zheng, and S. Li, "Automatic salient object segmentation based on context and shape prior," in *Proceedings of the British Machine Vision Conference (BMVC)*, vol. 3, 2011, p. 7.

[132] W. Zhu, S. Liang, Y. Wei, and J. Sun, "Saliency optimization from robust background detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014, pp. 2814–2821.

[133] H. Jiang, J. Wang, Z. Yuan, T. Liu, N. Zheng, and S. Li, "Salient object detection: A discriminative regional feature integration approach," in *International Journal of Computer Vision*, vol. 123, Springer, 2017, pp. 251–268.

[134] J. Ježek, T. Müller, Z. Materna, J. Dobransky, and M. Kociš, "Deep learning-based defect detection of metal parts: Evaluating current methods in complex conditions," *arXiv preprint arXiv:2103.03896*, 2021.

[135] P. Liznerski, L. Ruff, R. A. Vandermeulen, B. J. Franks, M. Kloft, and K.-R. Müller, "Explainable deep one-class classification," *arXiv preprint arXiv:2007.01760*, 2020.

[136] F. Di Mattia, P. Galeone, M. De Simoni, and E. Ghelfi, "DAGAN: Dual attention generative adversarial network for unsupervised anomaly detection," in *arXiv preprint arXiv:2102.07197*, 2021.

[137] Z. You, L. Cui, Y. Shen, *et al.*, "A unified model for multi-class anomaly detection," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 35, 2022, pp. 4571–4584.

[138] Y. Zou, J. Jeong, L. Pemula, D. Zhang, and O. Dabeer, "SPot-the-Difference self-supervised pre-training for anomaly detection and segmentation," in *European Conference on Computer Vision (ECCV)*, Springer, 2022, pp. 392–408.

[139] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *International Conference on Machine Learning (ICML)*, PMLR, 2020, pp. 1597–1607.

[140] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum contrast for unsupervised visual representation learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 9729–9738.

[141] X. Chen and K. He, "Exploring simple siamese representation learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021, pp. 15 750–15 758.

[142] X. Zhang, S. Li, X. Li, P. Huang, J. Shan, and T. Chen, "DeSTSeg: Segmentation guided denoising student-teacher for anomaly detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 3914–3923.

[143] V. Zavrtanik, M. Kristan, and D. Skočaj, "Dsr — a dual subspace reprojection network for surface anomaly detection," in *Computer Vision – ECCV 2022*, S. Avidan, G. Brostow, M. Cissé, G. M. Farinella, and T. Hassner, Eds., ser. Lecture Notes in Computer Science, Cham: Springer Nature Switzerland, 2022, pp. 539–554.

[144] Q. Zhou, W. Li, L. Jiang, *et al.*, "Pad: A dataset and benchmark for pose-agnostic anomaly detection," *Advances in Neural Information Processing Systems*, vol. 36, pp. 44 558–44 571, 2023.

[145] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "NeRF: Representing scenes as neural radiance fields for view synthesis," *Communications of the ACM*, vol. 65, no. 1, pp. 99–106, 2021.

[146] K. Zhou, X. Chang, T. Kim, *et al.*, "RAD: A Dataset and Benchmark for Real-life Anomaly Detection with Robotic Observations," *arXiv preprint arXiv:2410.00713*, 2024.

[147] J. Davis and M. Goadrich, "The relationship between precision-recall and roc curves," in *Proceedings of the 23rd International Conference on Machine Learning*, ser. ICML '06, Pittsburgh, Pennsylvania, USA: Association for Computing Machinery, 2006, pp. 233–240, ISBN: 1595933832. DOI: 10.1145/1143844.1143874. [Online]. Available: https://doi-org.ezproxy.unibo.it/10.1145/1143844.1143874.

[148] A. Borji, M.-M. Cheng, H. Jiang, and J. Li, "Salient object detection: A benchmark," *IEEE Transactions on Image Processing*, vol. 24, no. 12, 2015, ISSN: 1941-0042. DOI: 10.1109/tip.2015.2487833. [Online]. Available: http://dx.doi.org/10.1109/TIP.2015.2487833.