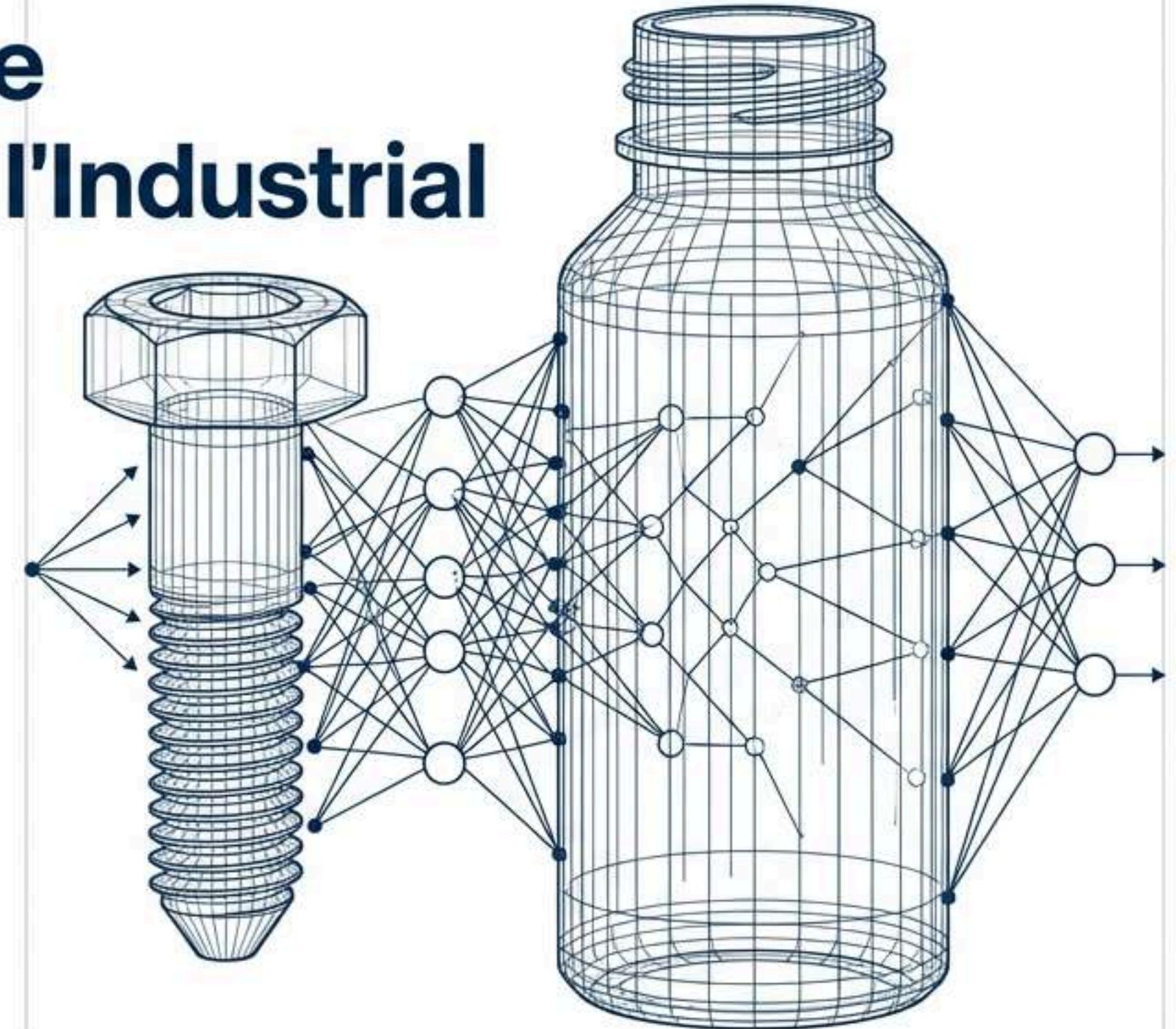


# Analisi e Ottimizzazione di SuperSimpleNet per l'Industrial Anomaly Detection

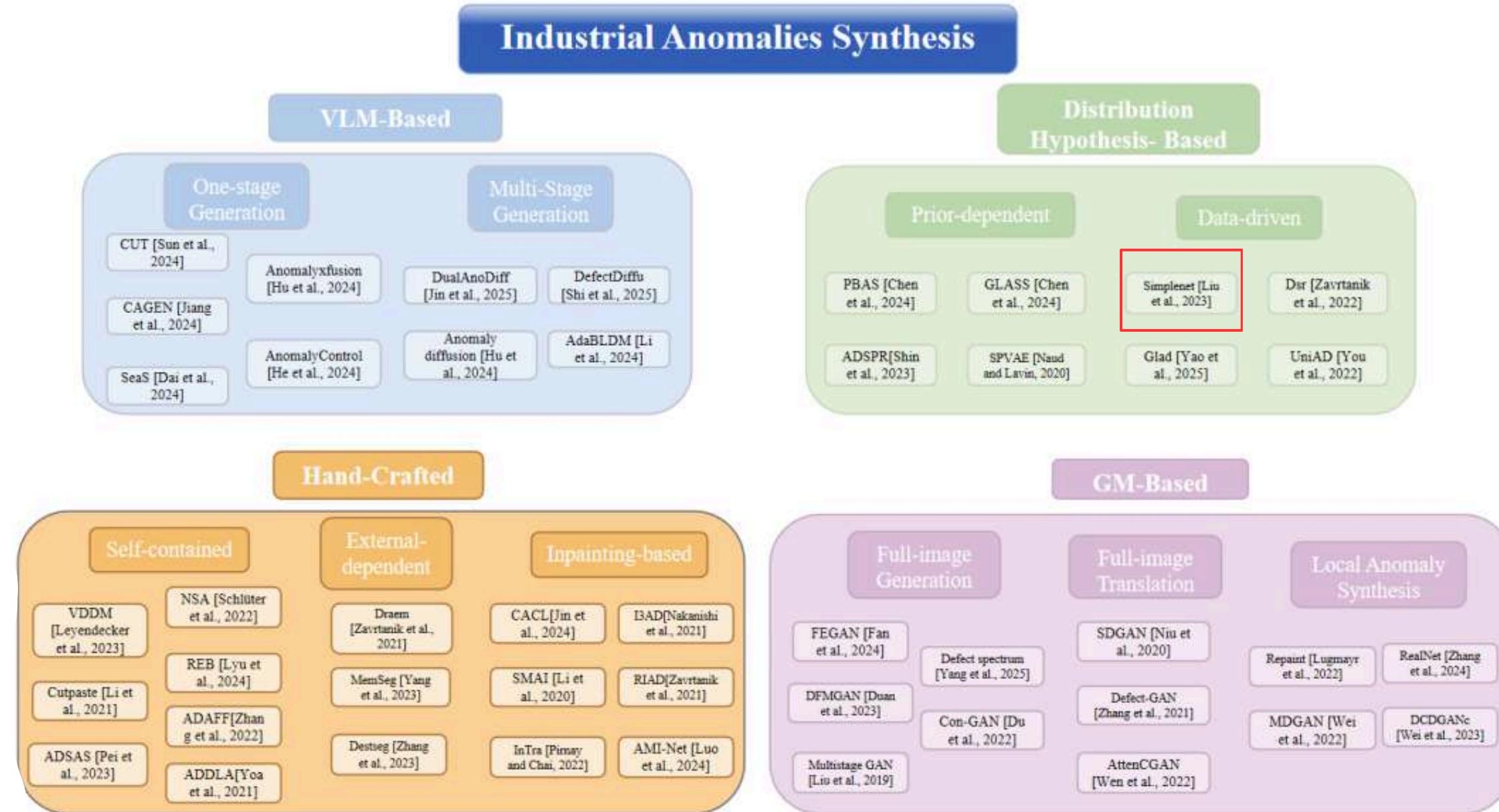
Efficienza, Robustezza  
e Supervisione Debole



**Giuseppe Zappia**  
Università della Calabria  
Anno Accademico 2025-2026

Obiettivo: Ottimizzazione resource-constrained (GPU Tesla P100)  
Metodologia: Sostituzione Backbone · Rumore Appreso · Weak Supervision

# La Sfida: Scarsità di Dati e Sintesi delle Anomalie



## Il Problema

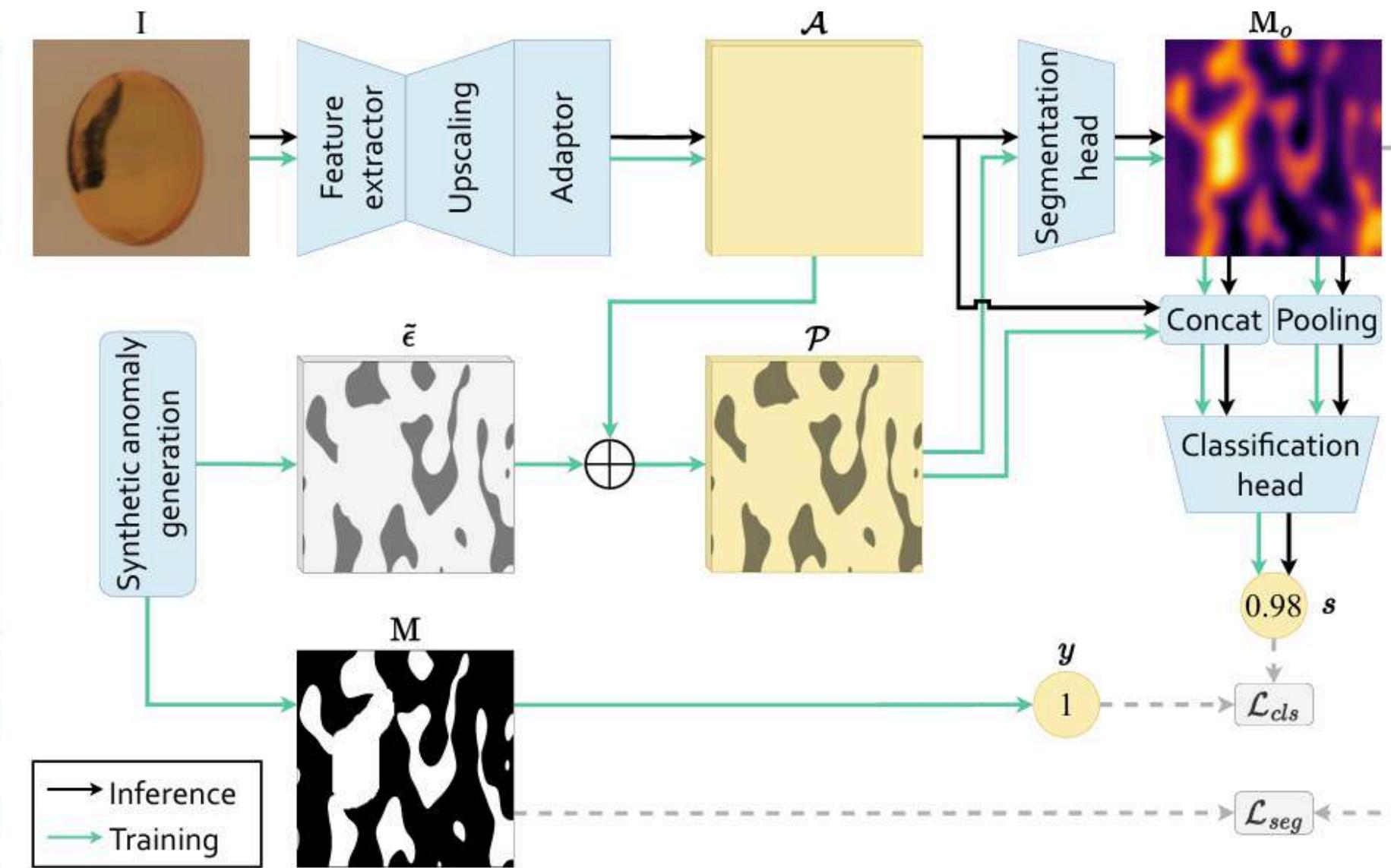
- I difetti reali sono rari e costosi da annotare.
- Soluzione: **Industrial Anomaly Synthesis (IAS)**. Generare campioni sintetici per l'addestramento.

## La Soluzione Scelta: SuperSimpleNet

- Approccio ibrido che opera nello spazio delle feature (Latent Space).
- Non genera immagini intere, ma perturba le feature estratte.
- Vantaggio: Efficienza dei metodi non supervisionati + precisione della segmentazione.

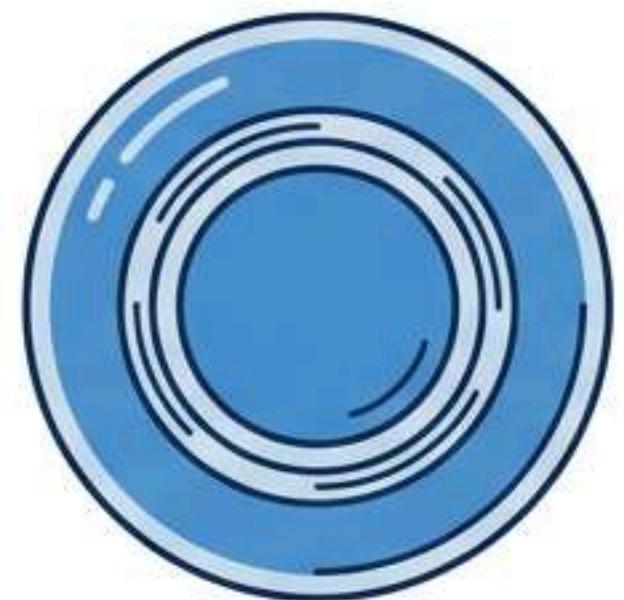
# L'Evoluzione Architetturale: Da SimpleNet a SuperSimpleNet

Caratteristica	SimpleNet	SuperSimpleNet
Risoluzione Feature	Bassa (Downsampling)	<b>Alta (Upscaling x2 e x4)</b>
Sintesi Anomalie	Rumore Gaussiano diffuso	<b>Maschere Perlin (Regioni contigue)</b>
Decisione	Discriminatore Singolo	<b>Dual Head (Segmentation + Classification)</b>



**Insight:** L'uso del Perlin Noise e dell'Upscaling permette di rilevare difetti strutturali mantenendo un'inferenza rapida (<10ms su V100).

# Metodologia Sperimentale e Vincoli



**Bottle (Oggetto Semplice)**

Test di efficienza pura.  
Difetti macroscopici.



**Carpet (Texture Complessa)**

Test di coerenza spaziale.  
Pattern ripetitivi.



**Screw (Stress Test)**

La sfida critica. Geometria rigida +  
Texture fine (filettatura).



Local Dev

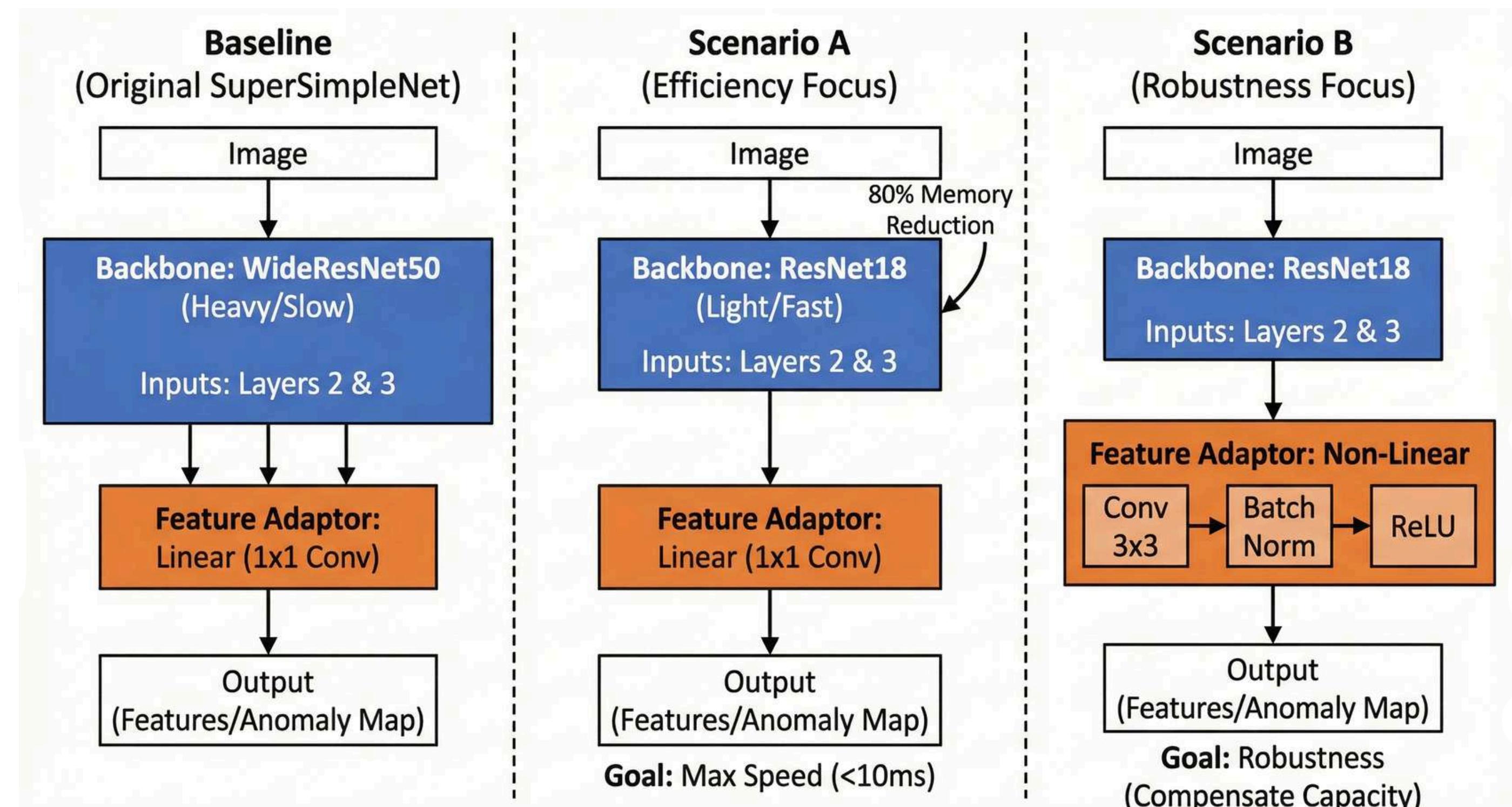


- **Piattaforma:** Kaggle con NVIDIA Tesla P100.
- **Obiettivo:** Replicare le performance della V100 (paper originale) su hardware meno potente tramite ottimizzazione.



Cloud Execution

# Modifica 1: Alla Ricerca dell'Efficienza (Backbone Swap)



**Ipotesi:** È possibile sostituire la pesante WideResNet50 con una ResNet18 mantenendo l'accuratezza?

# Tabella Comparativa dei Risultati sperimentali:

Categoria	Metrica	Paper Ufficiale (V100)	Baseline Riprodotta (P100)	Scenario A (ResNet18)	Scenario B (ResNet18 + Non-Lin)
BOTTLE	I-AUROC (Det)	100.0%	100.0%	100.0%	71.3%
	AUPRO (Loc)	90.4%	90.2%	86.8%	29.7%
CARPET	I-AUROC (Det)	98.4%	98.5%	89.6%	75.9%
	AUPRO (Loc)	92.3%	92.5%	86.3%	21.4%
SCREW	I-AUROC (Det)	92.9%	91.2%	51.1%	46.6%
	AUPRO (Loc)	95.3%	94.3%	71.0%	20.2%

La ResNet18 si rivela ottimale per geometrie semplici, ma la compressione delle feature espone i limiti del rumore statico su texture fini, e l'aggiunta di complessità (Scenario B) degrada ulteriormente la discriminazione

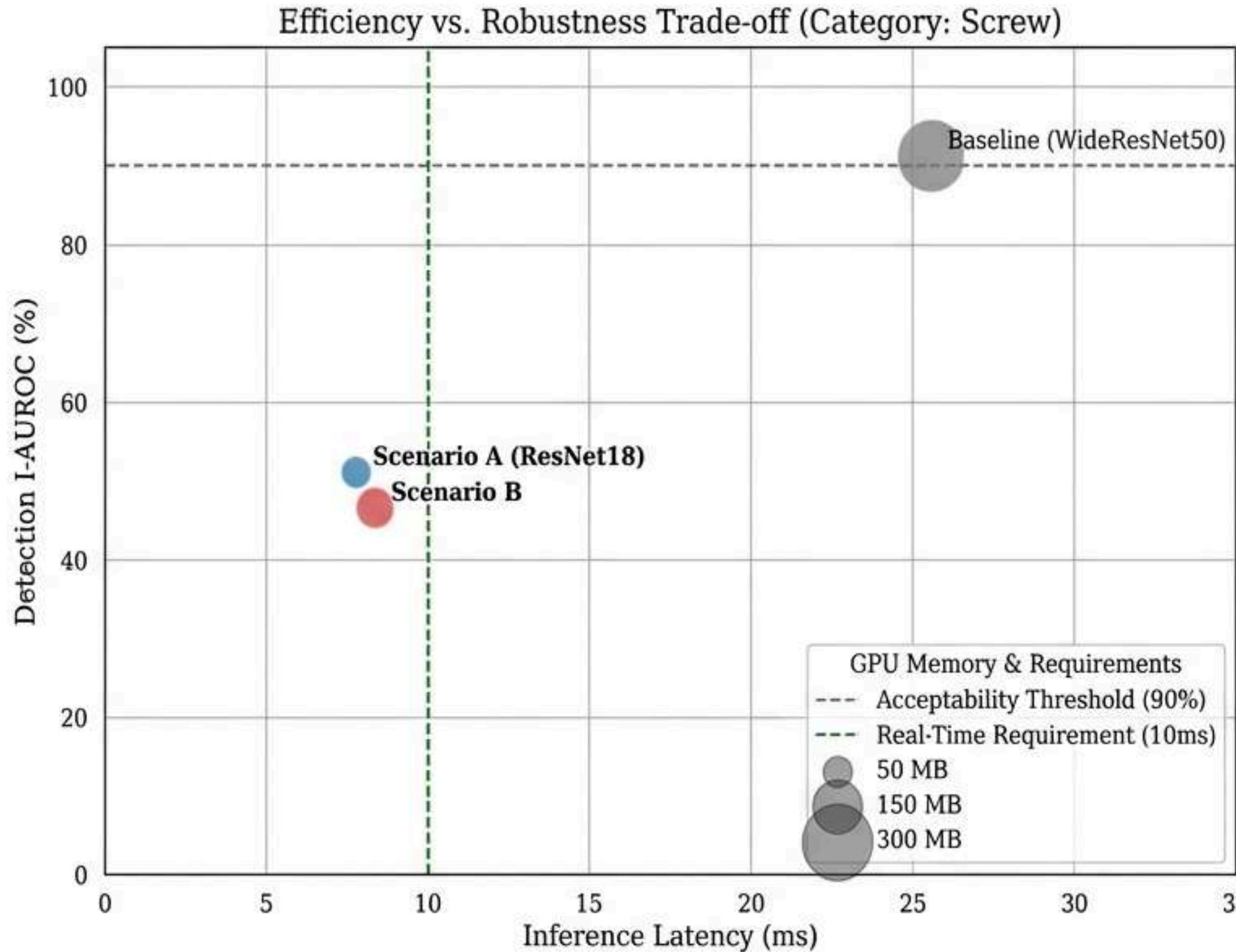
# Valutazione Benchmark di Efficienza Computazionale:

Configurazione	Parametri Totali	Inference Time (ms)	Throughput (img/s)	GPU Memory (MB)
Baseline (WideResNet50)	33.7 M	25.58 ms	96.72	270.53 MB
<b>Scenario A (ResNet18)</b>	<b>4.5 M</b>	<b>7.78 ms</b>	<b>398.47</b>	<b>52.43 MB</b>
Scenario B (ResNet18 + BN)	5.7 M	8.36 ms	352.69	85.98 MB

Lo Scenario A abbatte l'occupazione di memoria dell'80.6% e quadruplica il throughput, portando il tempo di inferenza sotto la soglia critica dei 10ms per l'ispezione in tempo reale

Lo Scenario B introduce invece un overhead di memoria del 64% senza apportare benefici prestazionali

# Risultati Modifica 1: Il Trade-off Velocità vs Accuratezza



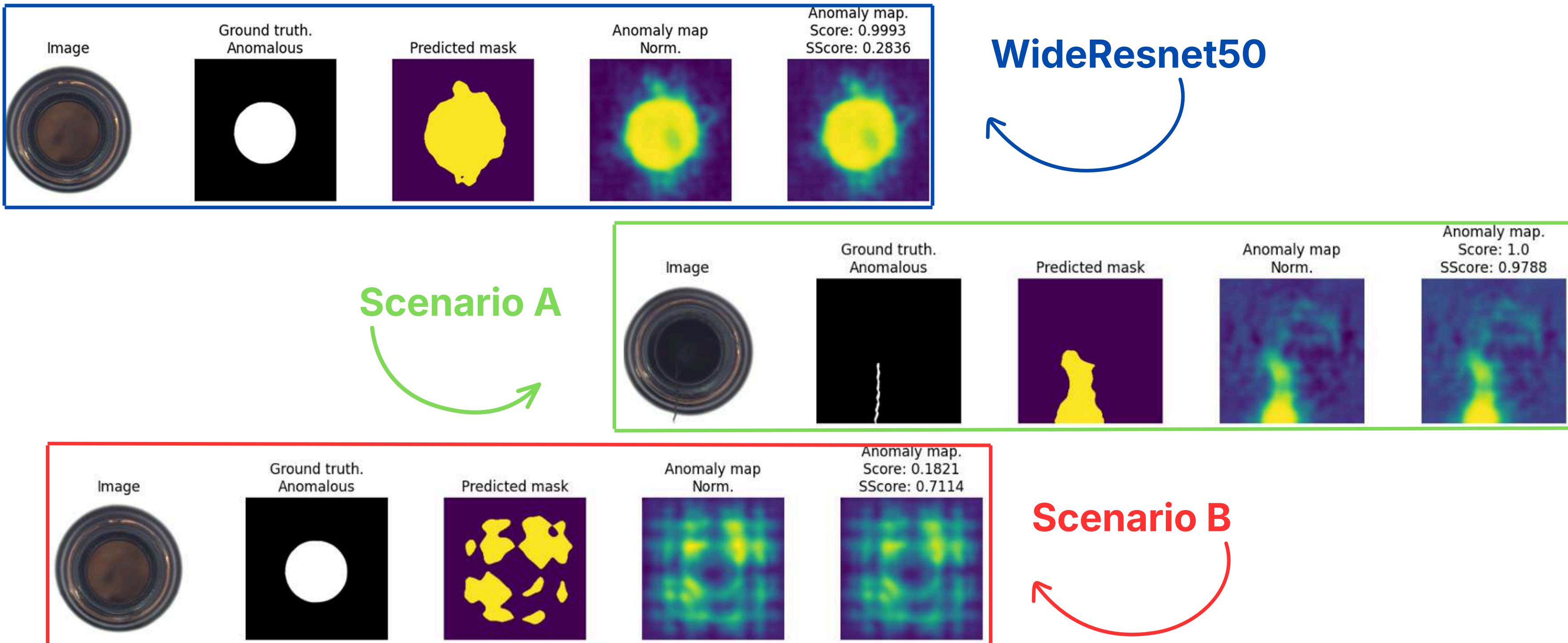
## Bottle vs. Screw

**Bottle:** 100% I-AUROC.  
Memoria **-80%** (52MB).  
Throughput **400 img/s.**

**Screw:** Accuratezza crolla al **51.1%** (Random Guessing).

La ResNet18 è veloce, ma cieca sulle texture fini (filettature).

# Analisi Visiva dei Risultati sulla Categoria Bottle



# Analisi del Fallimento: Perché la ResNet18 fallisce sulle Viti?



Allucinazione dei Difetti (Falsi Positivi diffusi)

## 1. Backbone Dependency:

La ResNet18 comprime troppo le feature. La "filettatura" della vite viene confusa con il rumore statico ( $\sigma=0.015$ ).

## 2. Il Paradosso dello Scenario

### B:

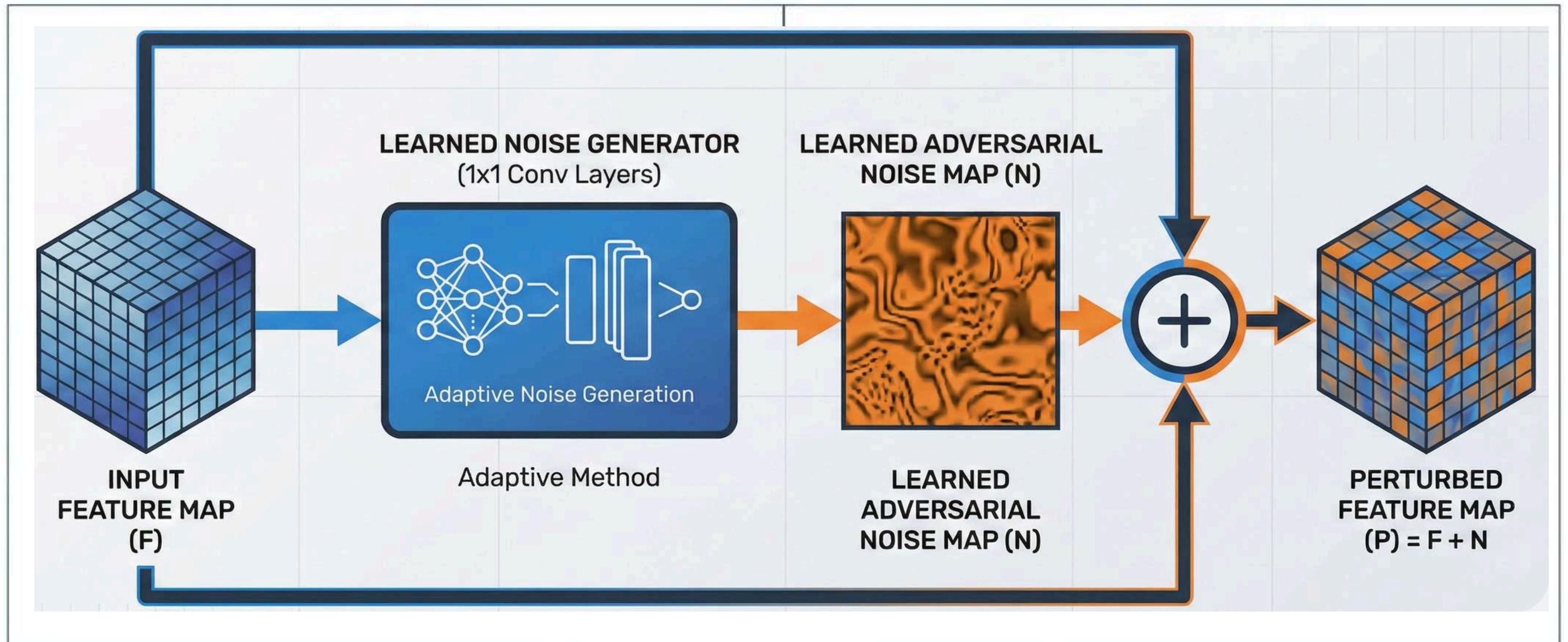
L'aggiunta di **Batch Normalization** ha amplificato la varianza naturale della texture.

## Conseguenza:

Il discriminatore non distingue più tra "texture vite" e "rumore sintetico".

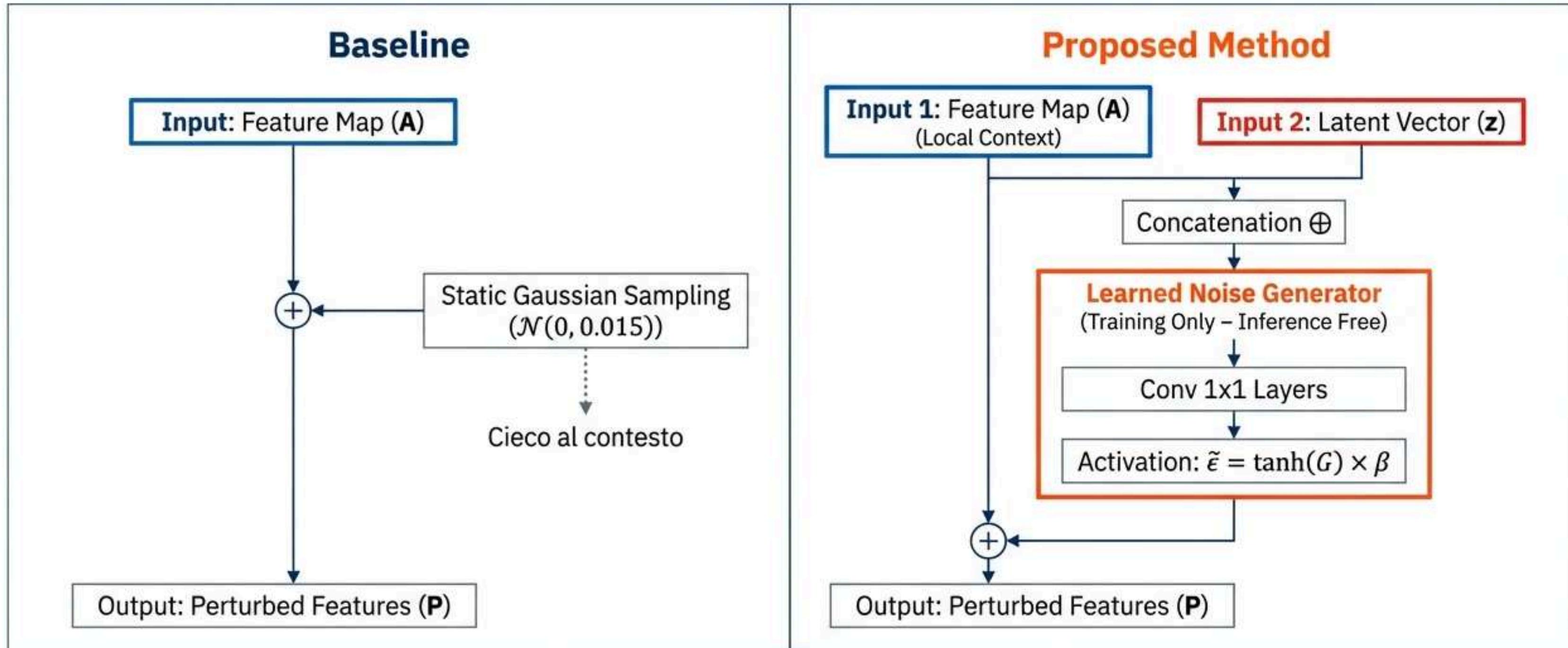
Risultato: I-AUROC 46.6%.

# Modifica 2: Generazione Adattiva del Rumore (v1)



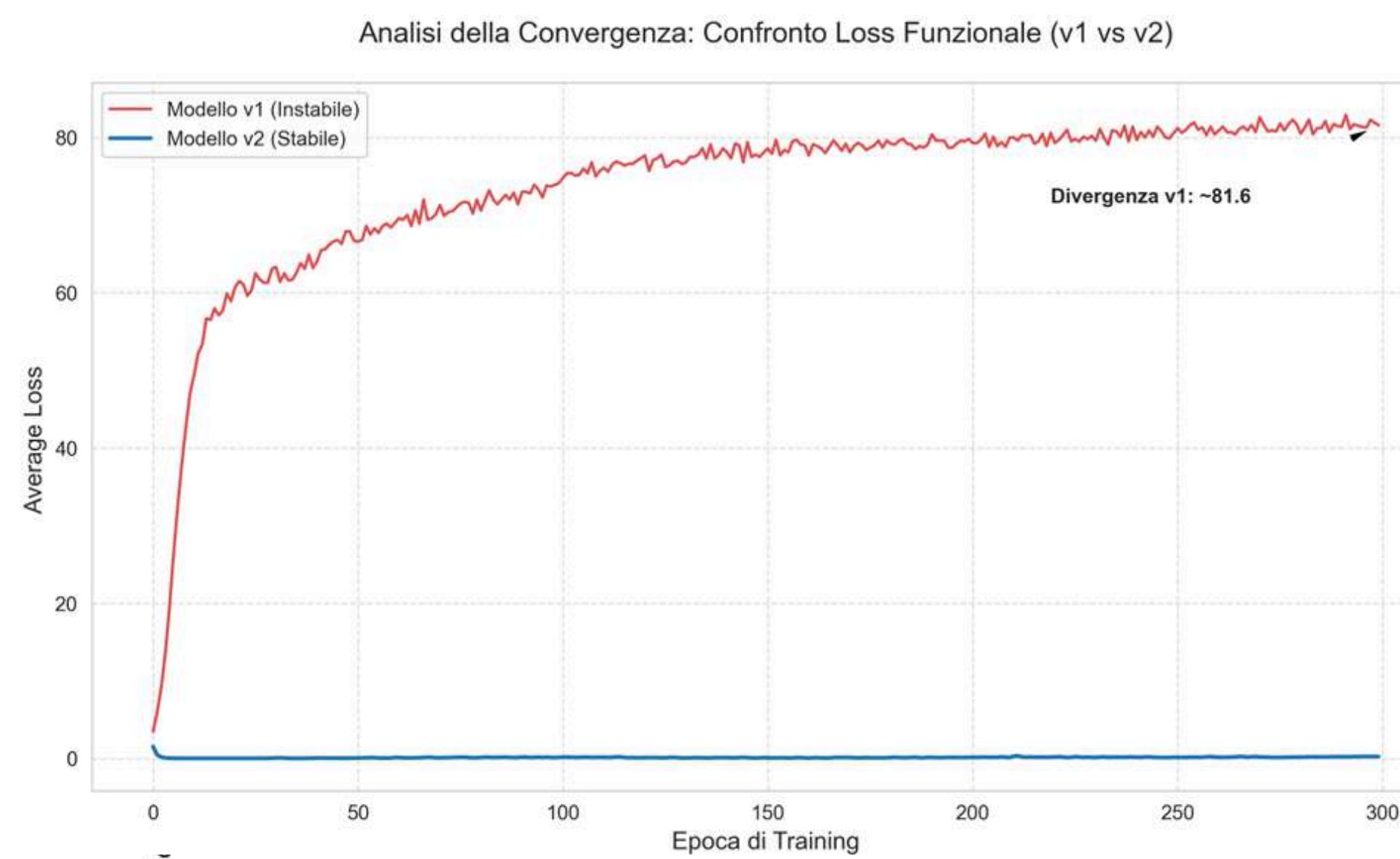
**Versione 1 Learned Noise Generator:** Implementazione di un paradigma avversoriale 'naive' che, privo di specifici vincoli, ha causato un'instabilità numerica estrema e il collasso della generalizzazione (I-AUROC 39.8%)

# Modifica 2: Generazione Adattiva del Rumore (v2)



**Versione 2 Learned Noise Generator:** Stabilizzazione del paradigma avversario tramite vincoli di magnitudo ( $\tanh, \beta$ ), regolarizzazione L2 e introduzione della stocasticità (vettore  $z$ ).

# Stabilizzazione del Training: Da v1 a v2



## Soluzione Tecnica (v2)

- Attivazione Tanh + Scaling ( $\beta$ ):** Limita la magnitudo del rumore ( $\tilde{\epsilon} = \tanh(G(A)) \times \beta$ ).
- Stocasticità (z vector):** Introduce variabilità casuale per prevenire la memorizzazione.
- Magnitude Constraint ( $L_{mag}$ ):** Penalizza perturbazioni troppo forti.
- Update Ratio 3:1:** Il discriminatore si aggiorna più spesso del generatore.

## Risultati Modifica 2: Il Recupero della Robustezza

Configurazione	Beta ( $\beta$ )	I-AUROC	P-AUROC	Note
Baseline (ResNet18)	-	51.1%	89.6%	Random Guessing
Mod 2 (High Noise)	0.5	46.8%	45.3%	Saturazione
<b>Mod 2 (Low Noise)</b>	<b>0.05</b>	<b>72.3%</b>	63.4%	<b>Best Detection</b>
<b>Mod 2 (Mid Noise)</b>	<b>0.1</b>	67.8%	<b>84.8%</b>	<b>Best Localization</b>
Mod 2 (Low Noise v2)	0.0015	48.4%	74.5%	Random Guessing

### Key Insight Box

#### Il Trade-off del Rumore:

- **Low Noise ( $\beta=0.05$ ):** Ideale per capire se c'è un difetto (Detection).
- **High Noise ( $\beta=0.1$ ):** Aiuta a trovare i bordi (Localization), ma introduce rumore nella classificazione globale.

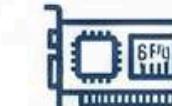
# Modifica 2: Discussione dei Risultati

Metrica 	Scenario A (Original Loss)	Learned Noise v1	Learned Noise v2 (Evoluto)
I-AUROC ↑	0.5113	0.3987	<b>0.6784</b>
AP-det ↑	0.7759	0.7163	<b>0.8268</b>
P-AUROC ↑	<b>0.8960</b>	0.5555	0.8481
AUPRO ↑	<b>0.7100</b>	0.2876	0.6185
AP-loc ↑	<b>0.1514</b>	0.0038	0.0213
seg-I-AUROC ↑	<b>0.8237</b>	0.4642	0.5544
seg-AP-det ↑	<b>0.9372</b>	0.7327	0.7795

## La Versione 2:

- ✓ stabilizza il training avversario
- ✓ migliora i risultati della detection su texture complesse
- ✗ introduce un trade-off che privilegia la classificazione globale rispetto alla precisione millimetrica della segmentazione.

# Modifica 2: Efficienza Computazionale Invariata



Configurazione	Speed (Latenza)	Throughput	GPU Memory
<b>Scenario A (Fixed Noise)</b>	<b>7.78 ms</b>	398.47 img/s	<b>52.43 MB</b>
<b>Scenario Learned Noise</b>	<b>7.83 ms</b>	396.58 img/s	<b>54.53 MB</b>
<i>Delta</i>	<i>+0.05 ms</i>	<i>-1.89 img/s</i>	<i>+2.10 MB</i>

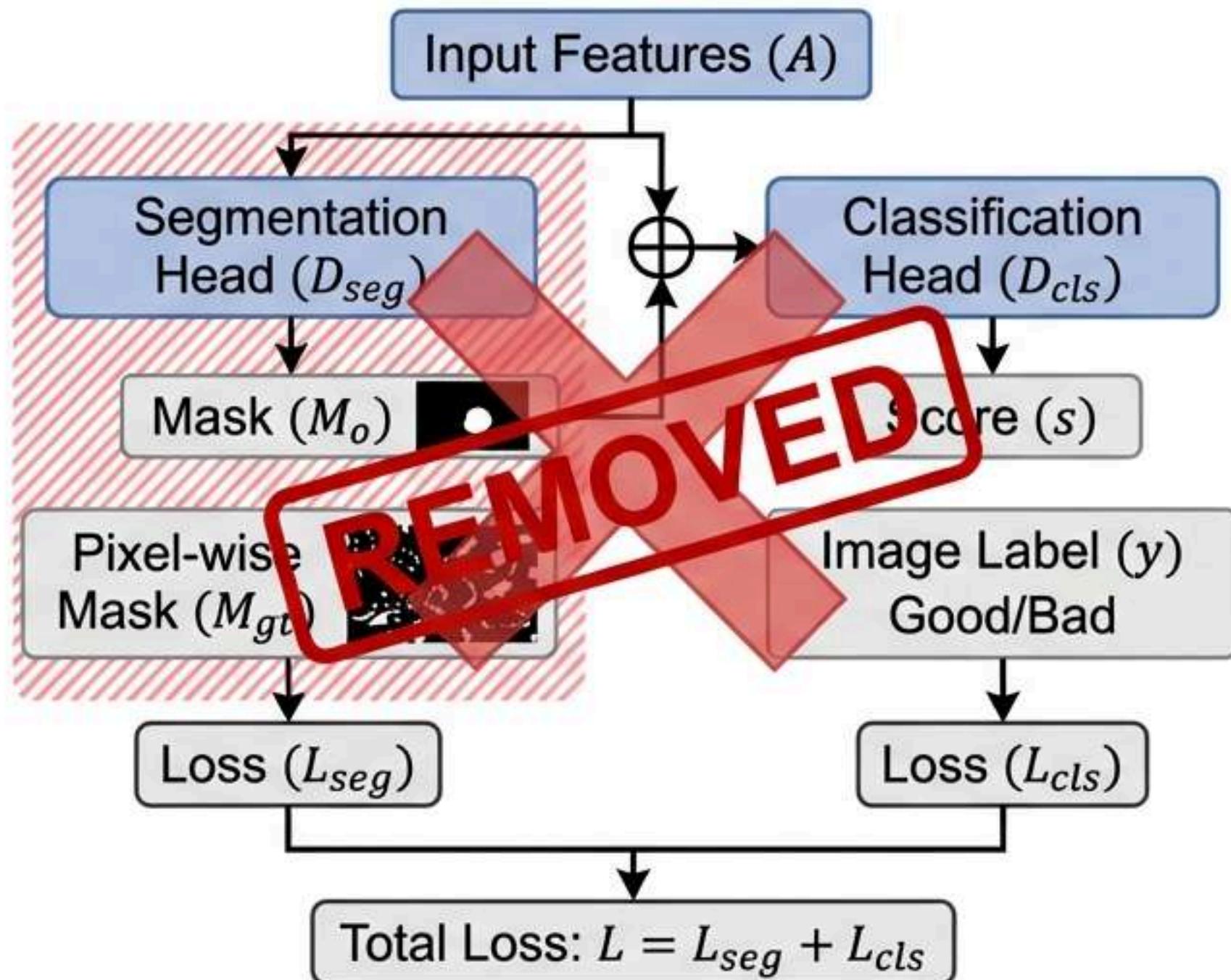
## EFFICIENZA DELLO SCENARIO A PRESERVATA:

L'introduzione del Learned Noise Generator preserva l'efficienza dello Scenario A grazie alla sua natura 'Inference-Free'. L'incremento di latenza e memoria è trascurabile , il modello garantisce la massima velocità industriale senza alcun overhead in fase di test

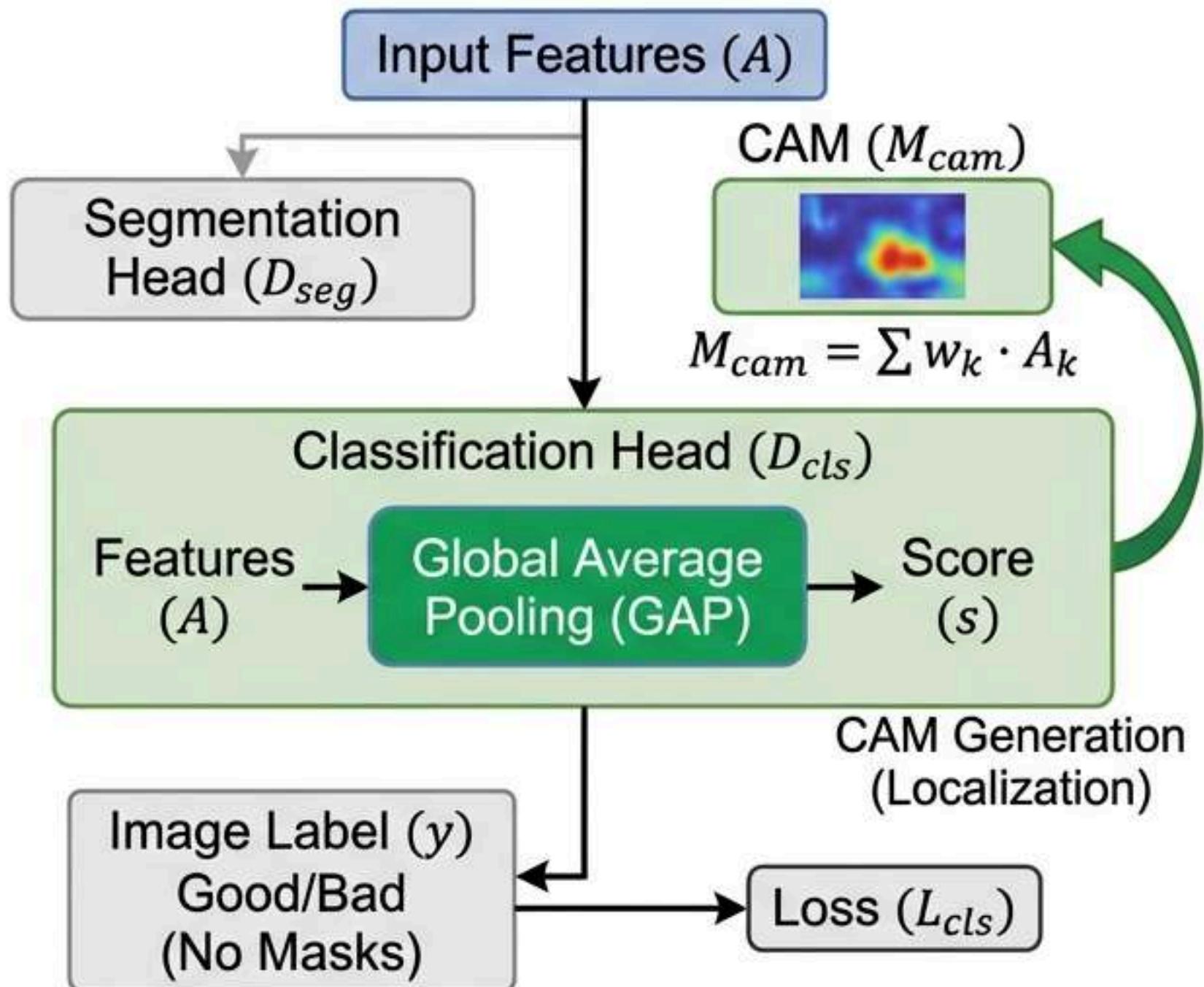


# Modifica 3: Scalabilità tramite Supervisione Debole

## Fully-Supervised



## Weakly-Supervised (GAP + CAM)



Requires ONLY Image Label (No Masks)

- **Obiettivo:** Addestrare usando solo etichette binarie.
- Dataset: **KSDD2** (Surface Defects).

# Modifica 3: Formulazione Matematica del CAM

## Score di Anomaly

$$s = \sum_{k=1}^C w_k \cdot \left( \frac{1}{H \times W} \sum_{i,j} A_{k,i,j} \right)$$

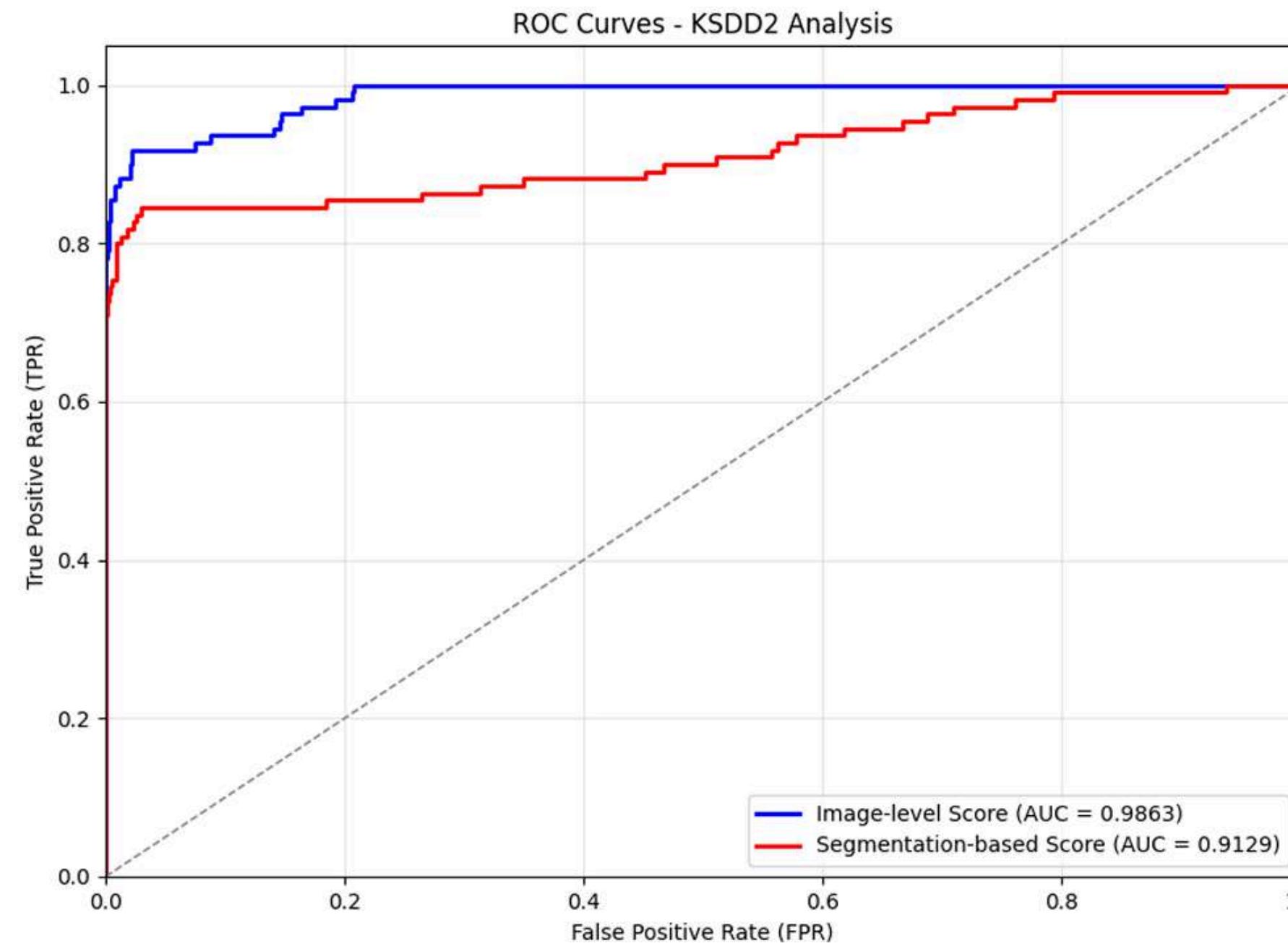
Calcolo globale del punteggio di anomalia (Inference)

## Mappa di Localizzazione

$$M_{cam}(i, j) = \sum_{k=1}^C w_k \cdot A_k(i, j)$$

Generazione della mappa di calore per la localizzazione

# Modifica 3: Performance

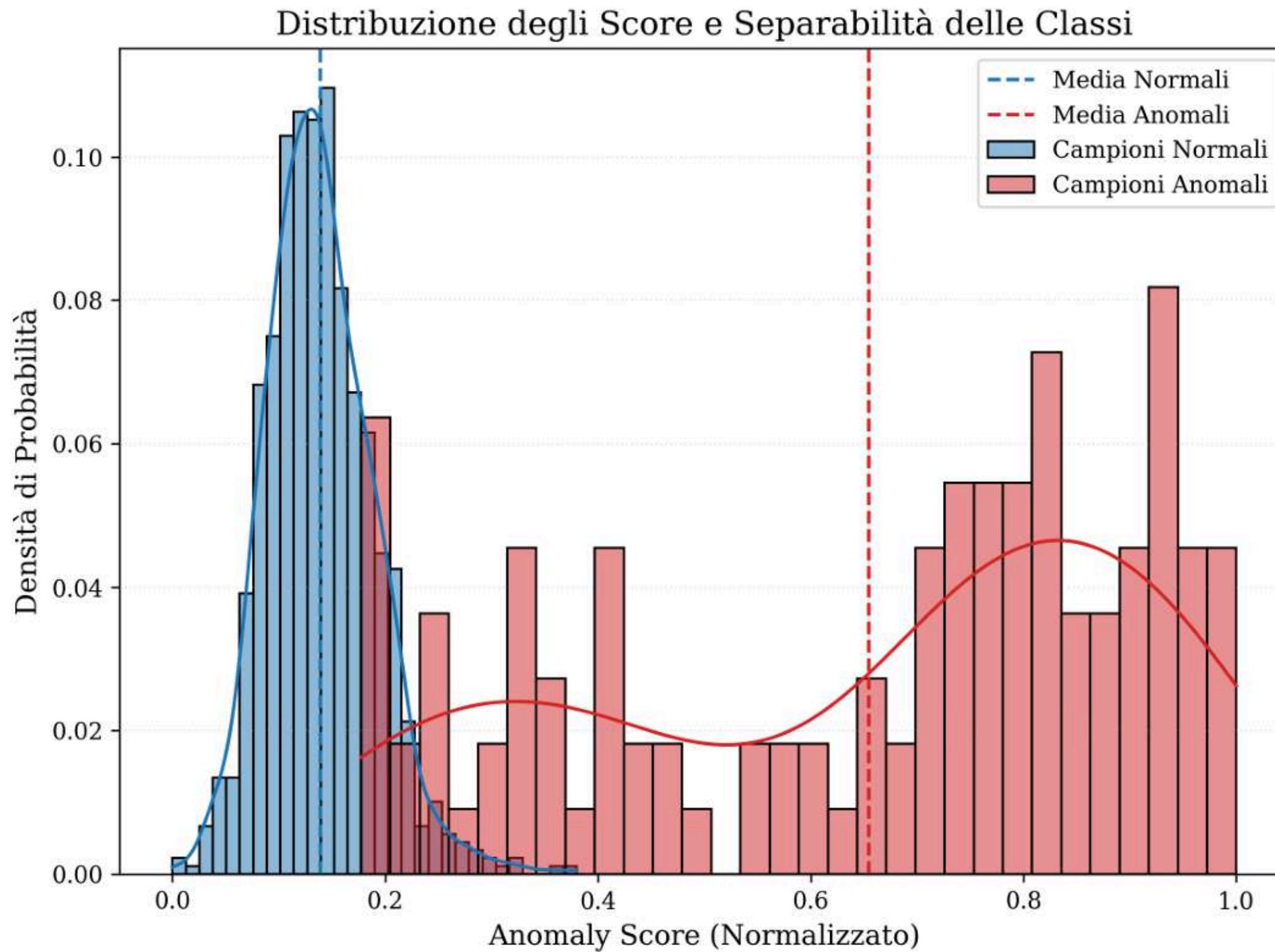


Detection Metrics		Localization Metrics	
I-AUROC	98.6%	P-AUROC	87.2%
AP-det	94.5%	AUPRO	87.4%
		AP-loc	7.4%

Nonostante la rimozione della supervisione tramite maschere, il classificatore è in grado di apprendere feature robuste per distinguere correttamente i campioni difettosi da quelli conformi

La soglia di decisione ottimale, tramite la massimizzazione dell'**indice di Youden** ( $J = TPR - FPR$ ) è identificato in **0.2489**

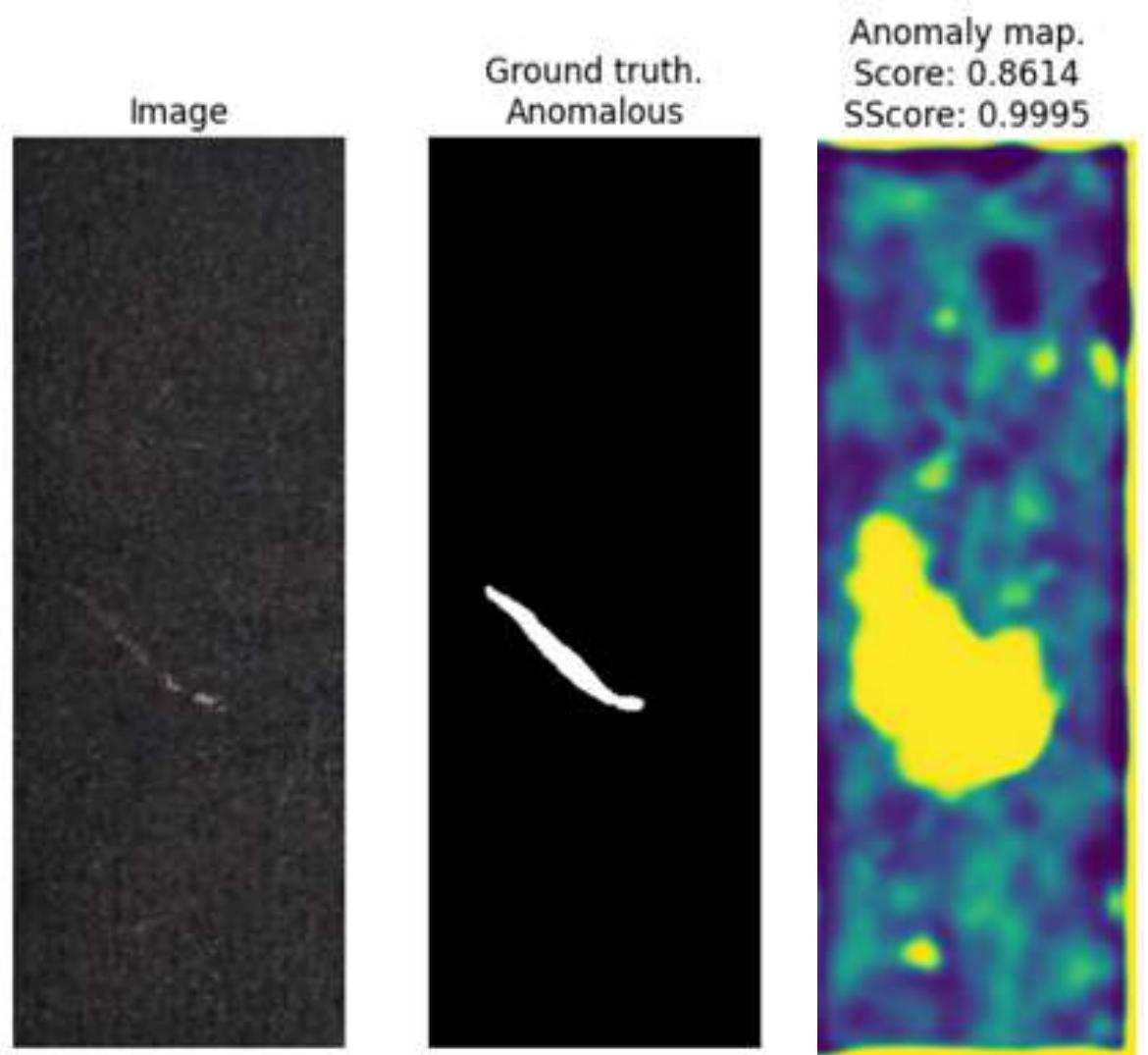
# Modifica 3 Risultati: Eccellenza nel Rilevamento



**I-AUROC: 98.6%**  
**AP-det: 94.5%**

- **Sensitivity (TPR):** 91.8%. Il modello identifica correttamente oltre 9 difetti su 10.
- **False Positive Rate:** 2.2%. Risultato eccellente per l'automazione industriale.

# Il Limite della Supervisione Debole: Il “Blob Effect”



**AUPRO: 87.4%**

(Il modello guarda nel posto giusto).

**AP-loc: 7.4%**

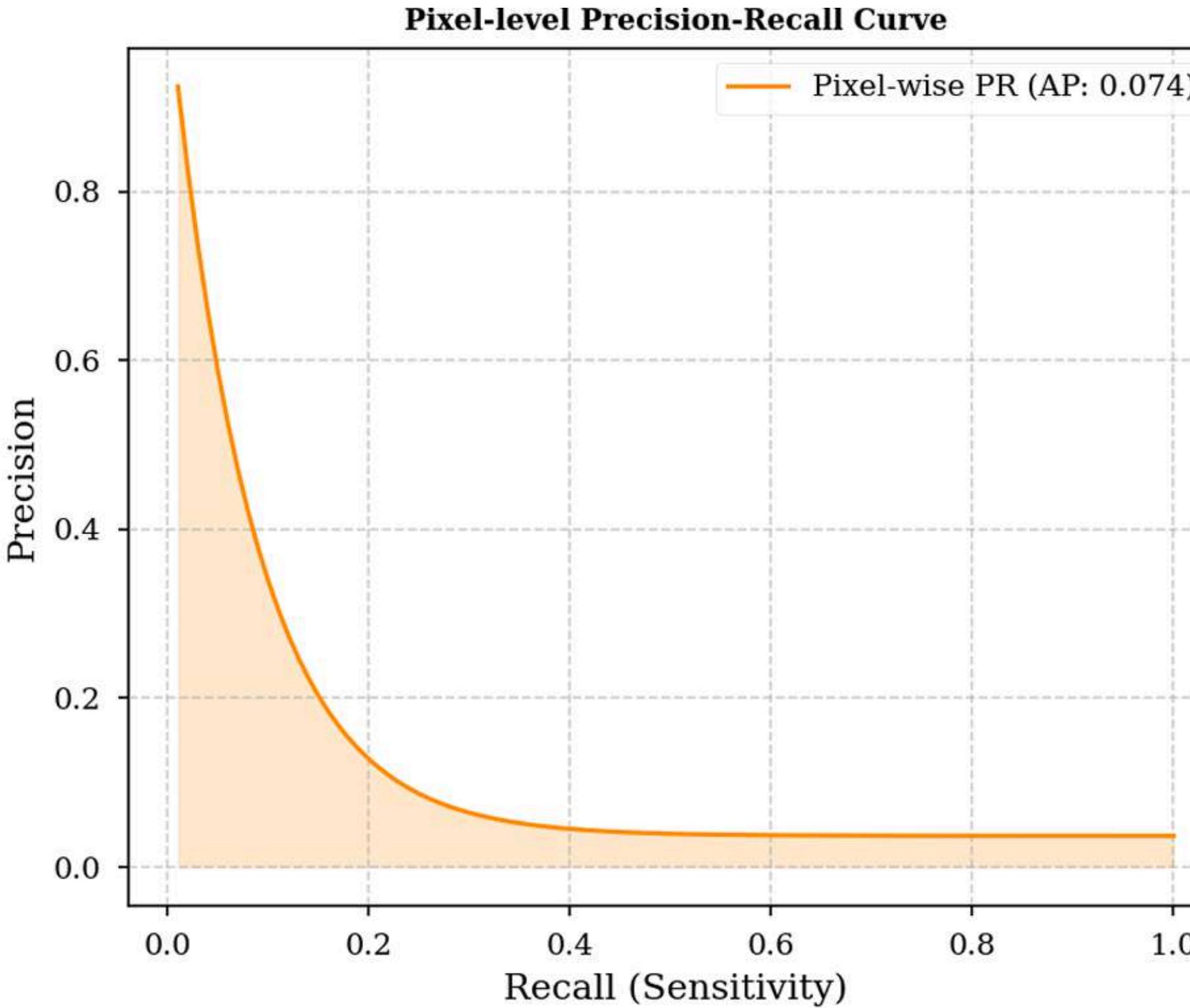
(La precisione dei bordi è bassa).

## Il Problema:

Le CAM hanno bassa risoluzione. Quando upscalate, diventano ‘macchie’ (blob).

Senza una loss di segmentazione pixel-wise ( $\mathcal{L}_{seg}$ ), la rete non ha incentivo a stringere la maschera attorno al graffio.

# Modifica 3: Localizzazione



- **Alta Precisione Iniziale:** La curva parte da valori vicini a 1.0, confermando che i picchi massimi di attivazione coincidono correttamente con i difetti reali
  - **Il "Blob Effect":** Crollo della precisione all'aumentare del Recall visto che le CAM a bassa risoluzione generano "macchie" diffuse che, per coprire l'intero difetto (graffi sottili), inglobano inevitabilmente lo sfondo sano
- L'AP di 0.074 non indica dunque un mancato rilevamento, ma misura la discrepanza tra la risoluzione grossolana della supervisione debole e la richiesta di precisione pixel-perfect**

# Sintesi Comparativa

Modello	Velocità	Costo Dati	Precisione Texture	Note
<b>Baseline</b> (WideResNet50)	Lenta	Alto (Maschere)	Alta	Precisa ma pesante.
<b>Mod 1</b> (ResNet18)	Altissima	Alto (Maschere)	Bassa (Fallimento)	Ottima solo per oggetti semplici (Bottle).
<b>Mod 2</b> (Learned Noise)	Altissima	Alto (Maschere)	Media/Buona	Recupera la robustezza senza costi di runtime.
<b>Mod 3</b> (Weakly-Sup)	Altissima	Zero (Solo Label)	Bassa (Blob)	Perfetta per la detection, meno per maschere precise

# Modifica 3: Confronto con SimpleNet e SuperSimpleNet

Metodo	Regime	AP-det (Det)	AP-loc (Loc)	ΔAP-det (vs Paper)
SimpleNet	Unsupervised	88.4%	89.6%	-9.0%
SuperSimpleNet	Fully-Supervised	97.4%	93%	/
Modificata (GAP+CAM)	Weakly-Supervised	94.5%	7.3%	-2.9%

# Conclusioni e Prospettive Future

## Sostenibilità Hardware:

ResNet18 è viabile per l'industria. Il Learned Noise è la chiave per mantenere l'accuratezza su hardware limitato (Inference-free).

## Riduzione Costi:

GAP+CAM elimina la necessità di dataset annotati pixel-perfect per i task di classificazione

## Prossimo Passo: Mixed Supervision:

Utilizzare un approccio ibrido: 90% dati Weakly-Supervised + 10% Maschere per raffinare i bordi e risolvere il ‘Blob Effect’ senza esplosione dei costi.