DIPARTIMENTO DI INGEGNERIA INFORMATICA, MODELLISTICA, ELETTRONICA E SISTEMISTICA

Corso di Laurea Magistrale in Ingegneria Informatica

Machine & Deep Learning

Relazione

Re-Engineering AE-XAD with Vision Transformers for Explainable Anomaly Detection

Professore:

Prof. **Fabrizio Angiulli**

Studente:

Presta Vincenzo

matr. 252290

ANNO ACCADEMICO 2024/2025

AE-XAD - ViT

Elaborato finale: Machina & Deep Learning

Vincenzo Presta

Novembre 2025

Indice generale

1 Introduzione

L'anomaly detection su immagini è un compito particolarmente complesso, poiché richiede non solo l'identificazione di osservazioni che deviano dal comportamento normale, ma anche la capacità di spiegare quali componenti dell'immagine giustifichino tale devianza. In numerosi contesti applicativi — come ispezione industriale, manutenzione predittiva o monitoraggio di qualità — non è sufficiente stabilire se un'immagine sia anomala: è necessario produrre una spiegazione interpretabile e coerente che evidenzi le regioni responsabili dell'anomalia.

La letteratura tradizionale sull'Explainable Artificial Intelligence (xAI) si è concentrata principalmente su modelli di classificazione o regressione, mentre una minore attenzione è stata dedicata al problema dell'Explainable Anomaly Detection (xAD). Tuttavia, le peculiarità del rilevamento di anomalie rendono i metodi post-hoc scarsamente efficaci in questo scenario [1]. In particolare, tre criticità strutturali ostacolano l'applicazione diretta delle tecniche di spiegabilità classiche:

- **Rarità delle anomalie:** i difetti sono poco rappresentati e non esiste un processo affidabile per generarli artificialmente in modo realistico; ciò limita l'uso di metodi post-hoc basati sull'abbondanza di esempi anomali.
- **Eterogeneità delle anomalie:** forme, estensioni e intensità possono variare significativamente rendendo difficile definire un modello universale.
- **Bassa fedeltà delle spiegazioni post-hoc:** tali metodi operano su modelli già addestrati, non progettati per essere esplicabili, producendo mappe che spesso non riflettono accuratamente le regioni realmente responsabili dell'anomalia.

A fronte di questi limiti, è cresciuto l'interesse verso approcci *explainability-by-design*, ossia modelli che integrano il meccanismo di spiegazione all'interno del processo di addestramento. In questo contesto si colloca **AE–XAD**, un metodo basato su Autoencoder che ha come obiettivo non solo il rilevamento dell'anomalia, ma anche la produzione di una *heatmap* interpretabile in grado di evidenziare le regioni che più contribuiscono al comportamento anomalo.

Il problema affrontato da AE–XAD può essere formulato come segue:

> *Data un'immagine potenzialmente anomala, determinare non solo se essa contenga difetti, ma anche quali regioni dell'immagine siano responsabili della deviazione rispetto al comportamento normale.*

I metodi ricostruttivi convenzionali, come gli Autoencoder standard, non sono sufficienti per questo scopo: tendono infatti a ricostruire anche le regioni anomale e a produrre mappe di errore poco informative e rumorose. AE–XAD supera tali limitazioni introducendo una strategia di addestramento semi-supervisionata progettata per:

- ricostruire accuratamente i pixel normali;

- ricostruire intenzionalmente *in modo errato* i pixel anomali;
- generare una heatmap stabile, coerente e interpretabile basata sull'errore di ricostruzione.

Nelle sezioni successive si analizzeranno i principi operativi del metodo, la formulazione matematica della loss, la struttura dell'architettura e il processo di generazione della heatmap.

2 Formulazione del problema

Sia un insieme di training $X = \{x_1, \ldots, x_n\}$, dove ciascuna osservazione $x_i \in [0,1]^D$ rappresenta un'immagine normalizzata. Per ogni immagine è inoltre disponibile una heatmap binaria $y_i \in \{0,1\}^D$, che specifica per ciascun pixel se esso sia normale ($y_{i,j} = 0$) oppure anomalo ($y_{i,j} = 1$). Come indicato in [1], una heatmap assume valore zero ovunque per gli esempi *inlier*, mentre presenta almeno un pixel attivo per gli esempi *outlier*.

La quantità $\|y_i\|_1$ (somma degli elementi della heatmap) permette di distinguere formalmente:

- gli **inlier**, caratterizzati da $\|y_i\|_1 = 0$;
- gli **outlier**, per cui $\|y_i\|_1 > 0$.

Sia inoltre I l'insieme degli indici degli inlier e O quello degli outlier, come definito nel paper.

Il compito del modello, dato un insieme di immagini di test $T = \{t_1, \ldots, t_m\}$, consiste nel generare per ciascuna immagine una heatmap h_i che stimi il contributo di ogni pixel all'anomalia complessiva. La heatmap può essere:

- **binaria**, con $h_{i,j} \in \{0,1\}$;
- **continua**, tipicamente $h_{i,j} \in [0,1]$, indicando il grado di outlierness del pixel.

Come descritto in [1], la dimensione del dato D coincide con $H \times W \times C$, ovvero le dimensioni spaziali e il numero di canali dell'immagine. L'obiettivo finale è definire per ogni immagine un valore di *anomaly score* $S(t_i)$, idealmente proporzionale alla presenza e all'estensione delle regioni anomale.

Il problema posto da AE–XAD può quindi essere formalizzato come segue:

> dato un insieme limitato di esempi anomali annotati a livello pixel, addestrare un modello capace di ricostruire fedelmente le regioni normali e di enfatizzare, attraverso la ricostruzione, le regioni anomale, così da ottenere una heatmap interpretabile che evidenzi i pixel maggiormente responsabili della devianza.

Questo scenario rientra nel paradigma della *semi-supervised anomaly detection*, poiché il modello sfrutta informazioni supervisionate limitate (le heatmap anomale sparse) e, al tempo stesso, apprende la distribuzione dei pixel normali da grandi quantità di dati privi di difetti.

3 Architettura del modello

L'architettura di AE–XAD è composta da un encoder convoluzionale pre-addestrato, da un decoder con struttura biforcata e da un modulo finale che combina le due ricostruzioni per ottenere un'immagine finale $\tilde{x}$. Come descritto in [1], la rete è progettata per guidare l'Autoencoder verso una ricostruzione accurata dei pixel normali e, al contempo, una ricostruzione intenzionalmente distante dei pixel anomali, secondo quanto imposto dalla loss semi-supervisionata.

3.1 Encoder

L'encoder è costruito selezionando i primi tre blocchi residuali di una rete ResNet pre-addestrata su ImageNet. Tale scelta consente di sfruttare un estrattore di feature robusto ed efficiente, capace di catturare strutture di basso e medio livello utili per la fase di ricostruzione. L'output della ResNet viene poi passato attraverso un ulteriore strato convoluzionale che riduce il numero di canali, producendo una rappresentazione latente di dimensione $28 \times 28 \times 64$. Come specificato nel paper, i pesi del modello ResNet rimangono congelati durante l'addestramento, al fine di ridurre il costo computazionale e preservare le capacità generalizzative dell'encoder pre-addestrato.

3.2 Decoder

Il decoder di AE–XAD è articolato in due rami paralleli, ciascuno con un ruolo distinto nel processo di ricostruzione dell'immagine.

Branch 1 (non-trainable). Questo ramo effettua un semplice upsampling della rappresentazione latente, da $28 \times 28 \times 64$ fino a $224 \times 224 \times 64$. Viene quindi applicata una funzione `tanh`, e successivamente i 64 canali vengono compressi tramite somma in gruppi da 8, producendo un tensore $b_1 \in \mathbb{R}^{224 \times 224 \times 8}$. Tale ramo funge da "maschera morbida" (*soft mask*) che enfatizza o attenua regioni specifiche della ricostruzione finale.

Branch 2 (trainable). Questo ramo comprende tre blocchi convoluzionali, ciascuno costituito da una convoluzione seguita da una trasposed convolution con attivazione SeLU. L'obiettivo è ricostruire progressivamente dettagli strutturali e texture dell'immagine, generando un tensore $b_2 \in \mathbb{R}^{224 \times 224 \times 8}$.

3.3 Combinazione dei due rami

Come riportato in [1], i due rami vengono combinati tramite la seguente operazione per-pixel:

$$b = b_1 \cdot b_2 + b_2.$$

Il termine b_1 agisce come una maschera adattiva che amplifica o sopprime particolari regioni di b_2. Questo meccanismo consente al decoder di focalizzare la ricostruzione nelle aree più informative, favorendo un errore di ricostruzione più marcato nelle regioni anomale durante l'addestramento.

Due ulteriori strati convoluzionali affiancati a valle di questa combinazione rifiniscono l'immagine ricostruita e convertono il tensore risultante in un output finale $\tilde{x} \in \mathbb{R}^{224\times224\times3}$, con la stessa dimensione dell'immagine di input.

3.4 Considerazioni architetturali

L'utilizzo di un encoder pre-addestrato e congelato riduce il costo computazionale e stabilizza l'addestramento, mentre la struttura biforcata del decoder permette di controllare con precisione la ricostruzione delle regioni anomale attraverso la loss specifica. Questa architettura rappresenta un compromesso efficiente tra capacità espressiva, interpretabilità e sostenibilità computazionale, come evidenziato dagli esperimenti presentati nel paper.

4 Loss Function

La componente centrale di AE–XAD è una loss semi-supervisionata progettata per ottenere una ricostruzione accurata delle regioni normali e, al tempo stesso, una ricostruzione deliberatamente distante nelle regioni anomale. Come mostrato in [1], dato un campione $x \in [0,1]^D$ e la relativa heatmap binaria $y \in \{0,1\}^D$, la loss per-pixel è definita come:

$$\ell(x,y) = \sum_{j=1}^{D} \left[(1-y_j)\,\frac{(x_j-\tilde{x}_j)^2}{(F(x_j)-x_j)^2} \;+\; \lambda_y\, y_j\, \frac{(F(x_j)-\tilde{x}_j)^2}{(F(x_j)-x_j)^2} \right]. \tag{1}$$

La loss totale sull'intero training set è data da:

$$L(X,Y) = \frac{1}{|X|} \sum_{(x,y)\in X\times Y} \ell(x,y). \tag{2}$$

4.1 Interpretazione dei termini

La formulazione della loss riflette due comportamenti distinti:

- **Pixel normali ($y_j = 0$):**

$$\frac{(x_j-\tilde{x}_j)^2}{(F(x_j)-x_j)^2}.$$

  In questo caso, il modello è incentivato a ricostruire fedelmente il pixel originale x_j. Il denominatore normalizza l'errore affinché i contributi siano comparabili tra diversi valori di intensità.

- **Pixel anomali ($y_j = 1$):**

$$\lambda_y \frac{(F(x_j) - \tilde{x}_j)^2}{(F(x_j) - x_j)^2}.$$

Qui il modello non deve ricostruire x_j, ma deve avvicinare $\tilde{x}_j$ al valore trasformato $F(x_j)$, forzando un errore di ricostruzione elevato nelle regioni anomale. Questo comportamento è essenziale affinché la differenza $|x_j - \tilde{x}_j|$ diventi un indicatore affidabile di anomalia.

4.2 Ruolo della trasformazione F

La funzione $F : [0, 1] \to \mathbb{R}$ è un iperparametro introdotto per massimizzare la distanza di ricostruzione nelle regioni anomale. Due scelte discusse in [1] sono:

- $F_{-}(x) = 1 - x$, che rappresenta il negativo del pixel;
- $F_v(x) = v$, con $v \notin [0, 1]$ (nel paper viene utilizzato $v = 2$).

In entrambi i casi, $F(x_j)$ è costruita per essere distante dall'originale x_j, aumentando così l'ampiezza dell'errore ricostruttivo nelle zone anomale.

4.3 Peso di bilanciamento λ_y

Il coefficiente λ_y compensa lo sbilanciamento tra pixel normali e anomali:

$$\lambda_y = \begin{cases} D/\|y\|_1 & \text{se } \|y\|_1 > 0, \\ 1 & \text{altrimenti.} \end{cases}$$

Poiché le anomalie sono generalmente molto rare nello spazio dei pixel, λ_y evita che la loss sia dominata dai termini relativi ai pixel normali.

4.4 Effetto complessivo della loss

Complessivamente, la loss in Eq. (1) realizza un comportamento *explainability-by-design*: le regioni normali vengono ricostruite con accuratezza, mentre quelle anomale vengono ricostruite in modo intenzionalmente errato. La differenza tra immagine originale e ricostruita diventa così una stima affidabile dell'outlierness locale, utilizzabile per generare heatmap interpretabili.

5 Training Strategy

L'addestramento di AE–XAD segue una strategia semi-supervisionata che combina data augmentation mirata e oversampling, con l'obiettivo di compensare la scarsità e l'eterogeneità delle anomalie nel dataset. Come proposto in [1], la pipeline di training si articola in due componenti principali.

5.1 Data augmentation sulle anomalie

Poiché gli esempi anomali sono poco rappresentati e spesso molto diversi tra loro, AE–XAD applica un potenziamento artificiale dei campioni anomalie prima dell'addestramento. In particolare, per ogni immagine anomala x nel training set:

- essa viene replicata 5 volte senza modifiche;
- ulteriori 10 copie vengono generate tramite una procedura di *copy–paste*: la regione anomala viene ritagliata e incollata su immagini normali, dopo trasformazioni geometriche standard (zoom, rotazioni, traslazioni, ecc.).

Questa strategia migliora la diversità delle anomalie osservate dal modello e rafforza la sua capacità di generalizzare a difetti di forma e dimensione variabili.

5.2 Oversampling all'interno del batch

Dopo l'augmentation, ogni batch B viene bilanciato imponendo la seguente proporzione:

$$\frac{1}{3}|B| \quad \text{anomalie,} \qquad \frac{2}{3}|B| \quad \text{inlier.}$$

Questa scelta contrasta l'effetto predominante dei pixel normali nella loss e garantisce che il modello riceva un segnale supervisionato sufficiente durante l'ottimizzazione, senza sovraccaricare il training con esempi anomali artificiali.

5.3 Ottimizzazione

AE–XAD viene addestrato per 200 epoche utilizzando l'ottimizzatore Adam con learning rate pari a 10^{-3} e weight decay pari a 10^{-4}, come riportato in [1]. La ResNet dell'encoder è mantenuta congelata durante tutto l'addestramento, mentre solo i livelli convoluzionali del decoder vengono aggiornati.

6 Inference e generazione delle heatmap

Durante la fase di inference, AE–XAD utilizza l'errore di ricostruzione per individuare le regioni anomale dell'immagine. Il processo, descritto in [1], si articola in tre passaggi principali: calcolo dell'errore, normalizzazione e filtraggio tramite una finestra gaussiana adattiva.

6.1 Errore di ricostruzione

Dato un test sample t, si calcola la ricostruzione $\tilde{t}$ ottenuta dall'Autoencoder e si definisce il vettore di errore:

$$e = (t - \tilde{t})^2.$$

L'errore contiene valori elevati nelle regioni in cui il modello non ha tentato di replicare fedelmente l'input, tipicamente corrispondenti alle zone anomale.

6.2 Normalizzazione

Per rendere l'errore confrontabile attraverso pixel e immagini, AE–XAD applica la normalizzazione seguendo la procedura definita in Eq. (3):

$$\tilde{e} = \frac{e - \mu_e}{\sigma_e},$$

dove μ_e e σ_e sono media e deviazione standard dei valori in e. La mappa normalizzata $\tilde{e}$ funge da *raw heatmap*, evidenziando in modo preliminare le regioni devianti.

6.3 Selezione automatica del filtro gaussiano

Per migliorare la coerenza spaziale della heatmap, viene applicato un filtro gaussiano F_k di dimensione $(2k+1)\times(2k+1)$. La dimensione k non è fissata a priori, ma viene stimata automaticamente da AE–XAD sulla base delle proprietà geometriche delle anomalie. In particolare [1]:

1. si considera la mappa binarizzata di $\tilde{e}$ utilizzando la soglia $\mu_{\tilde{e}} + \sigma_{\tilde{e}}$;
2. si analizzano in tale mappa le componenti connesse per stimare l'estensione media delle regioni anomale, orizzontalmente e verticalmente;
3. il valore k viene scelto come metà della maggiore tra tali estensioni.

In questo modo, il filtro si adatta alle dimensioni effettive del difetto, evitando sia oversmoothing su anomalie piccole sia dispersione su anomalie estese.

6.4 Heatmap filtrata e binarizzazione

Applicando il filtro gaussiano, si ottiene la heatmap finale:

$$h = F_k(\tilde{e}).$$

Se è richiesta una segmentazione binaria delle anomalie, la heatmap viene ulteriormente sogliata usando:

$$\text{threshold} = \mu_h + \sigma_h,$$

dove μ_h e σ_h sono media e deviazione standard dei valori in h.

6.5 Anomaly score

AE–XAD definisce un anomaly score più robusto rispetto alla norma dell'errore grezzo, sfruttando la coerenza spaziale enfatizzata dal filtro:

$$S(t) = \|e \cdot F_k(e)\|.$$

Questa formulazione privilegia regioni in cui l'errore è consistente e spazialmente continuo, riducendo l'impatto del rumore isolato e migliorando la capacità di rilevare difetti piccoli ma strutturati.

7 Motivazioni e obiettivi del progetto

Il metodo AE–XAD, presentato in [1], ha dimostrato di raggiungere risultati allo stato dell'arte nella rilevazione di anomalie industriali, sia a livello di classificazione (X-AUC) sia a livello di localizzazione (IoU, PRO). La combinazione di un encoder ResNet, un decoder convoluzionale e una procedura di filtraggio adattivo consente di produrre heatmap coerenti e stabili, rendendo AE–XAD una soluzione matura e competitiva.

Tuttavia, l'intero framework rimane fortemente dipendente dalla capacità espressiva dell'encoder, responsabile di estrarre le rappresentazioni su cui opera il decoder. Nel lavoro originale, tale encoder è una rete convoluzionale pre-addestrata. Negli ultimi anni, i Vision Transformer (ViT) hanno mostrato una capacità superiore nel modellare relazioni non locali e nel catturare strutture globali dell'immagine, grazie al meccanismo di attenzione multi-testa.

A partire da queste considerazioni, il nostro progetto non mira a modificare la logica di AE–XAD, ma a rispondere a una domanda specifica:

> *È possibile migliorare la qualità delle mappe di anomalia e la bontà delle ricostruzioni sostituendo l'encoder CNN di AE–XAD con un Vision Transformer pre-addestrato, mantenendo invariata l'intera pipeline?*

L'obiettivo del progetto è dunque duplice:

1. **Integrare un encoder ViT all'interno di AE–XAD** senza alterare il decoder, lo scoring e il processo di generazione delle heatmap.

2. **Valutare sperimentalmente l'impatto della sostituzione dell'encoder** su ricostruzione, localizzazione delle anomalie e metriche globali, confrontando i risultati con la versione originale basata su CNN.

Questa analisi consente di verificare se l'impiego di un backbone Transformer, ormai standard in molte applicazioni di visione, possa portare benefici anche nelle pipeline di anomaly detection basate su ricostruzione, preservando al tempo stesso la semplicità e l'interpretabilità del framework AE–XAD.

8 Vision Transformer come encoder

Negli ultimi anni, i Vision Transformer (ViT) hanno introdotto un cambio di paradigma nell'elaborazione delle immagini, mostrando prestazioni rilevanti in numerosi compiti di visione artificiale rispetto ai modelli convoluzionali tradizionali. L'idea centrale, introdotta in [2], è trattare un'immagine come una sequenza di patch e applicare il meccanismo di *self-attention* tipico dei Transformer originariamente sviluppati per l'elaborazione del linguaggio naturale.

Questa strategia consente al modello di catturare relazioni globali tra regioni dell'immagine, rendendo la rappresentazione latente più ricca e potenzialmente più adatta ai compiti di ricostruzione e localizzazione delle anomalie.

8.1 Patch embedding

Un'immagine $x \in \mathbb{R}^{H \times W \times 3}$ viene suddivisa in patch non sovrapposti di dimensione $P \times P$. Ogni patch è poi linearizzato e proiettato in uno spazio latente di dimensione D tramite una trasformazione lineare:

$$z_0^i = E(x_i), \qquad i = 1, \ldots, N,$$

dove $N = \frac{HW}{P^2}$ è il numero di patch e E è una proiezione lineare. Il risultato è una sequenza di embedding che rappresenta un'immagine come una struttura simile a una frase.

8.2 Positional encoding

Poiché i Transformer non possiedono una nozione intrinseca di struttura spaziale, è necessario aggiungere informazioni sulla posizione delle patch. Ad ogni embedding z_0^i viene aggiunto un positional encoding $f_{\text{pos}}(i)$:

$$\tilde{z}_0^i = z_0^i + f_{\text{pos}}(i).$$

In questo modo, il modello può distinguere patch identiche in posizioni diverse e apprendere relazioni strutturali tra di esse.

8.3 Self-attention

Il cuore del Transformer è il meccanismo di *multi-head self-attention* (MHSA), che permette a ogni patch di comunicare con tutte le altre:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right) V.$$

Applicando questo meccanismo su più teste parallele, il modello apprende simultaneamente più tipi di relazioni tra patch distanti, ottenendo una rappresentazione globale dell'immagine più espressiva rispetto alle CNN, che si basano su operazioni locali.

8.4 Benefici attesi come encoder in AE–XAD

L'integrazione di un encoder ViT all'interno del framework AE–XAD è motivata da tre proprietà fondamentali:

1. **Cattura di dipendenze globali**: il self-attention consente di modellare relazioni tra regioni distanti, migliorando la coerenza globale della ricostruzione.

2. **Rappresentazioni semantiche più ricche**: ViT tende a produrre embedding più discriminativi delle CNN pre-addestrate, favorendo la separazione tra regioni normali e anomalie.

3. **Generalità e adattabilità**: l'encoder ViT può essere sostituito senza modificare il decoder di AE–XAD, mantenendo la pipeline invariata e rendendo l'approccio compatibile sia con modelli pre-addestrati sia con tecniche auto-supervisionate come MAE [3].

Queste considerazioni forniscono le basi per l'integrazione di un encoder Transformer all'interno di AE–XAD, con l'obiettivo di valutarne l'impatto sulla ricostruzione e sulla generazione delle heatmap.

9 Integrazione del Vision Transformer in AE–XAD

9.1 Obiettivo

L'obiettivo del progetto è integrare un encoder basato su Vision Transformer (ViT) all'interno del framework AE–XAD mantenendo invariata la struttura del decoder e l'intera pipeline di generazione delle heatmap. L'idea centrale consiste nel sostituire l'estrattore di feature convoluzionale originale con un backbone Transformer pre-addestrato, lasciando inalterati tutti i meccanismi ricostruttivi e il processo di scoring.

9.2 Sostituzione dell'encoder

L'encoder convoluzionale viene rimpiazzato da un modello ViT-B/16 pre-addestrato su ImageNet, sfruttandone la proiezione a patch e i blocchi Transformer. L'immagine di input (224×224) viene suddivisa in patch 16×16, proiettata in uno spazio latente di dimensione 768, e successivamente elaborata dai blocchi di attenzione. Il risultato è una sequenza di 196 token, rimappata in una struttura spaziale $14 \times 14 \times 768$, coerente con la griglia prodotta dal ViT.

9.3 Ricostruzione della griglia spaziale

Poiché il decoder originale di AE–XAD richiede una mappa latente di dimensione $28 \times 28 \times 64$, è stato introdotto un modulo di ricostruzione spaziale che si occupa di:

1. **ridurre** la dimensionalità dei 768 canali tramite una sequenza di convoluzioni che preservano la risoluzione 14×14;
2. **ricostruire** la griglia 28×28 tramite una trasposed convolution che porta i canali a 64, ottenendo una rappresentazione perfettamente compatibile con il decoder.

Questa fase consente di integrare il ViT senza alterare l'architettura ricostruttiva del modello originale.

9.4 Decoder

Il decoder di AE–XAD è stato mantenuto integralmente identico alla formulazione del paper. Esso comprende:

- un primo ramo non addestrabile, basato su upsampling diretto, attivazione `tanh` e riduzione dei canali;
- un secondo ramo addestrabile composto da tre blocchi convoluzionali con trasposed convolutions;
- una fase di modulazione finale che combina i due rami tramite un'operazione per-pixel;
- un modulo finale che produce l'immagine ricostruita tramite una sigmoide.

L'intero processo ricostruttivo rimane quindi invariato, garantendo che qualsiasi differenza nelle heatmap o nelle metriche sia imputabile unicamente al cambiamento dell'encoder.

9.5 Pipeline finale

La pipeline complessiva del modello può essere descritta come:

$$x \rightarrow \text{Encoder ViT} \rightarrow \text{Proiezione CNN} \rightarrow \text{Decoder AE–XAD} \rightarrow \tilde{x},$$

dove il Transformer assume il ruolo di estrattore di feature, mentre la ricostruzione e la generazione dell'errore seguono esattamente la logica originale del framework AE–XAD. Questo approccio permette di valutare in modo isolato il contributo delle rappresentazioni prodotte dal ViT alla qualità della ricostruzione e alla capacità di individuare e localizzare le anomalie.

References

[1] F. Angiulli, F. Fassetti, L. Ferragina, and S. Nisticò, "Explaining anomalies through semi-supervised autoencoders," *Array*, 2025, dIMES, University of Calabria.

[2] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," in *International Conference on Learning Representations (ICLR)*, 2021.

[3] K. He, X. Chen, S. Xie, Y. Li, P. Dollár, and R. Girshick, "Masked autoencoders are scalable vision learners," *CVPR*, 2022.