Alma Mater Studiorum · Università di Bologna

DIPARTIMENTO DI INFORMATICA

Progetto d'esame

Classificazione binaria su Elliptic++

Blockchain and Cryptocurrencies

Presentata da:
Benedetta Bottari
Claudia Brunetti
Milena Mazza

Anno Accademico 2024/2025

Indice

Introduzione

Nel contesto della crescente attenzione verso la sicurezza delle criptovalute, l'analisi delle transazioni blockchain riveste un ruolo cruciale per l'identificazione di attività illecite. In questo studio, ci concentriamo sull'analisi del dataset Elliptic++, un'estensione del noto dataset Elliptic, utilizzato per tracciare le transazioni di Bitcoin e identificare comportamenti sospetti.

Il lavoro si articola in tre fasi principali. In primo luogo, conduciamo un'analisi statistica approfondita del dataset, focalizzandoci sulle distribuzioni delle feature principali. Questo ci permette di ottenere una comprensione più chiara della rete.

Successivamente, eseguiamo una Social Network Analysis (SNA) per comprendere la struttura e le dinamiche della rete di transazioni. Questo passo consente di evidenziare nodi chiave, componenti e indagare la struttura della rete.

Infine, procediamo all'addestramento di modelli di Graph Neural Networks (GNN) utilizzando diverse architetture, tra cui GraphSAGE, Graph Attention Networks (GAT) e modelli basati su Transformer. L'obiettivo è identificare transazioni sospette sfruttando le capacità di apprendimento strutturale di queste reti neurali.

Questo approccio integrato mira a migliorare l'accuratezza e l'efficacia dei modelli di identificazione, contribuendo a sviluppare strumenti più robusti per l'analisi delle transazioni blockchain e la prevenzione di attività fraudolente.

Capitolo 1

Dataset

1.1 Struttura

Il dataset Elliptic++[7] è un'estensione del dataset originale di Elliptic [6], progettato per migliorare l'analisi delle transazioni Bitcoin e l'identificazione di attività illecite sulla blockchain. Questo dataset amplia il precedente dataset Elliptic, includendo non solo informazioni sulle transazioni, ma anche sui portafogli (wallet) e sulle interazioni tra indirizzi.
Il dataset, progettato attraverso l'uso di dati strutturati a grafo, è suddiviso in due componenti principali:

- **Dataset delle Transazioni**: Fornisce dettagli sulle transazioni Bitcoin. Un nodo nel grafico rappresenta una transazione; un edge può essere visto come un flusso di Bitcoin tra una transazione e l'altra. Ogni nodo ha 166 caratteristiche ed è stato etichettato come lecito (1), illecito (2) o sconosciuto(3).
- **Dataset degli Attori** (Indirizzi Wallet): Offre informazioni sugli indirizzi dei portafogli Bitcoin e sulle loro interazioni. È composto da indirizzi wallet, consentendo il rilevamento di indirizzi illeciti (attori) nella rete Bitcoin, sfruttando i dati del grafico.

Le due componenti del dataset sono interconnesse infatti le transazioni coinvolgono uno o più indirizzi come input e output. Analizzando congiuntamente le informazioni sulle transazioni e sugli indirizzi, è possibile tracciare il flusso di Bitcoin attraverso la rete, identificare schemi di comportamento e individuare potenziali attività illecite.

Caratteristiche	
Nodes (transactions)	203,769
Edges (money flow)	234,355
Time steps	49
Illicit (class-1)	4,545
Licit (class-2)	42,019
Unknown (class-3)	157,205
Features	183

Tab. 1.1: Elliptic++ Transactions Dataset (Addresses)

Caratteristiche	
Wallet addresses	822,942
Nodes (temporal interactions)	1,268,260
Edges (addr-addr)	2,868,964
Edges (addr-tx-addr)	1,314,241
Time steps	49
Illicit (class-1)	14,266
Licit (class-2)	251,088
Unknown (class-3)	557,588
Features	56

Tab. 1.2: Elliptic++ Actors Dataset (Wallet Addresses)

Il dataset delle transazioni (TX) di Elliptic++ presenta una struttura molto simile a Elliptic; fornisce una visione dettagliata delle operazioni avvenute sulla blockchain di Bitcoin. Ogni transazione è rappresentata come un nodo in un grafo diretto, mentre gli archi rappresentano il flusso di denaro tra le transazioni. Il dataset contiene 203.769 transazioni con 234.355 connessioni tra di esse.
Ogni nodo è descritto da 183 feature, che è possibile suddivise in tre categorie principali:

- Caratteristiche locali
- Caratteristiche aggregative
- Caratteristiche generali

Il dataset degli attori, o wallet dataset, di Elliptic++ si concentra sugli indirizzi Bitcoin e sulle loro interazioni. Le etichette associate ai wallet, indicando tre categorie: illeciti (class-1), leciti (class-2) e sconosciuti (class-3), che permettono di identificare portafogli sospetti, mentre le feature dei wallet sono utili per analizzare le loro interazioni e studiare il comportamento degli indirizzi e individuare pattern fraudolenti.

Gli archi rappresentano le connessioni tra transazioni e indirizzi suddividendoli in due tipologie:

- Interazioni wallet-wallet: Rappresentano transazioni dirette tra due indirizzi.
- Interazioni wallet-transazione: Descrivono il flusso di Bitcoin da un indirizzo a una transazione e viceversa.

1.2 Statistical e Social Network Analysis

1.2.1 Dataset degli transazioni

Il dataset TX contiene informazioni sulle transazioni Bitcoin, organizzate in 166 feature e suddivise in 49 istanze temporali (time steps), ciascuno corrispondente a circa due settimane. Le feature sono divisibili in tre categorie principali:

- Local Features (93): Riguardano attributi interni alla transazione, come gli importi e i dettagli della transazione stessa.
- Aggregate Features (72): Riguardano attributi aggregati, derivati da transazioni correlate o vicine temporalmente.
- Generali: tra cui è compresa la classe target (class)

Distribuzione Temporale: Le transazioni sono organizzate in 49 time steps, Questo permette di analizzare l'evoluzione temporale delle attività sospette.

Statistical Analysis

Al fine di comprendere in modo più approfondito la struttura del dataset, è stata condotta un'analisi statistica preliminare sul dataset delle transazioni.
Tale analisi ha permesso di evidenziare alcune caratteristiche chiave delle variabili: in particolare per le colonne *Local_feature* e *Aggregate_feature* si è osservato che entrambe le categorie sono state standardizzate, tale evidenza emerge dalla media dei valori prossima allo zero e dalle deviazioni standard che si avvicinano a 1.

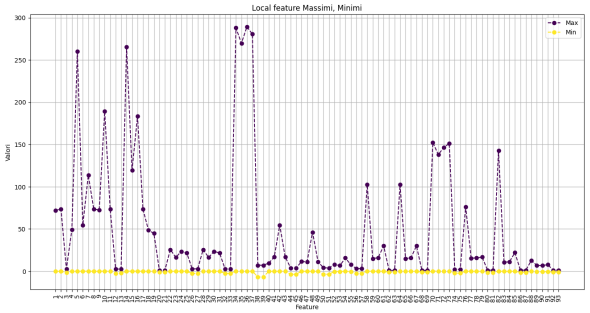

Fig. 1.1: Metriche minime e massime delle local feature

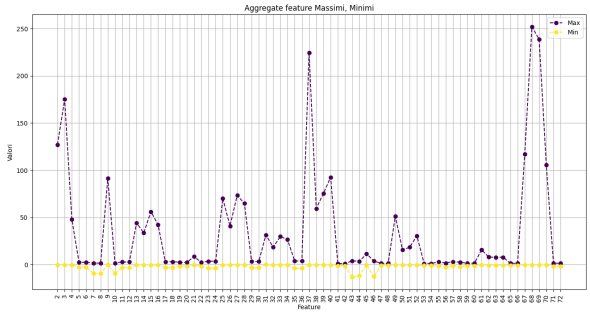

Fig. 1.2: Metriche minime e massime delle aggregate feature

L'analisi dei valori minimi e massimi ha evidenziato che i valori minimi di *Local_feature* variano in range prossimi allo 0, tra -6.99 e -0.01, mentre i valori massimi presentano una maggiore variabilità, con un range che va da -0.88 fino a 289.20. Analogamente per le *Aggregate_feature*, i valori minimi si distribuiscono in un intervallo ristretto compreso tra -13.09 e -0.08, mentre i valori massimi si collocano tra 1.06 e 251.84. Questa variabilità dei valori massimi suggerisce che alcune transazioni potrebbero avere comportamenti anomali o valori estremi.

Successivamente la ricerca si è concentrata sull'individuazione di eventuali pattern o correlazioni tra le colonne, con l'obiettivo di distinguere chiaramente tra la classe *legal* e *illegal*. A tale scopo, è stata calcolata la matrice di correlazione utilizzando l'indice di Pearson, confrontando separatamente i risultati ottenuti per le istanze appartenenti alla classe *legal* e *illegal*.

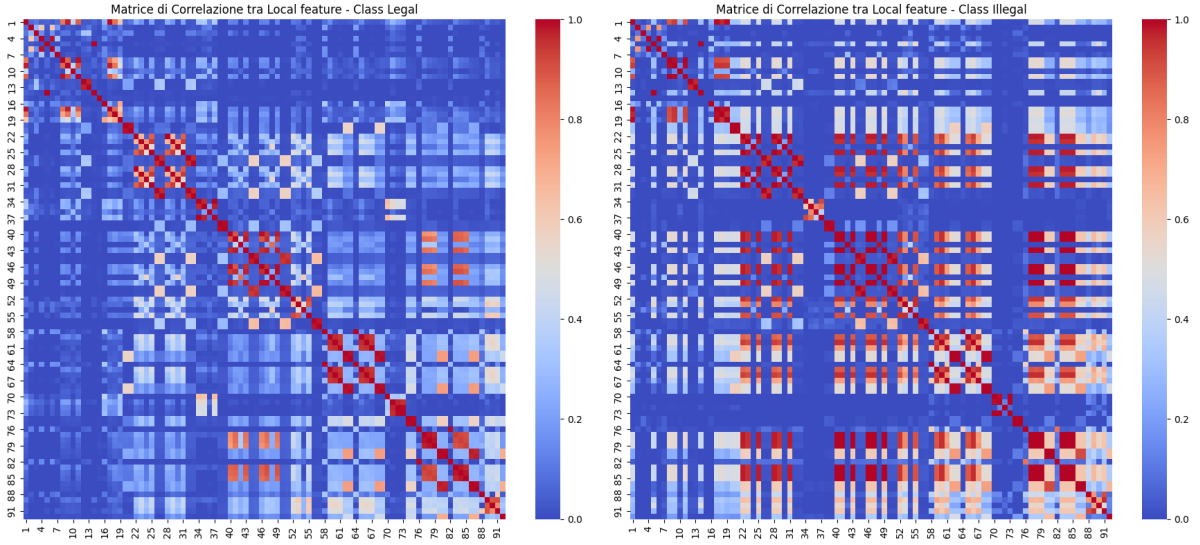

Fig. 1.3: Matrice di Correlazione tra Local feature di tx sulle classi illegal (1) e legal (2)

L'analisi delle correlazioni ha evidenziato una differenza significativa tra le due classi: i dati appartenenti alla classe *illegal* presentano correlazioni più elevate rispetto alla classe *legal*, in particolare nelle *Local Features* (figura 1.3). Questo fenomeno potrebbe suggerire la presenza di pattern specifici nella classe *illegal*; tuttavia, è più probabile che tale risultato sia dovuto alla minore numerosità dei dati *illegal* rispetto a quelli *legal*, il che porta a una distribuzione più ristretta e, di conseguenza, a correlazioni più alte tra le variabili.
Inoltre, sia l'analisi delle *Local Features* (fig. 1.3) sia quella delle *Aggregate Features* (fig. 1.4) ha rivelato la presenza di una struttura a blocchi nella matrice di correlazione,

indicando l'esistenza di gruppi di feature fortemente correlate tra loro. Questo fenomeno suggerisce che alcuni pattern ripetitivi potrebbero riflettere una ridondanza informativa o una dipendenza tra le variabili. Tale struttura a blocchi evidenzia la possibilità di applicare tecniche di riduzione della dimensionalità, come la selezione delle feature o l'Analisi delle Componenti Principali (PCA), al fine di migliorare l'efficienza dei modelli senza compromettere la capacità descrittiva delle caratteristiche analizzate.

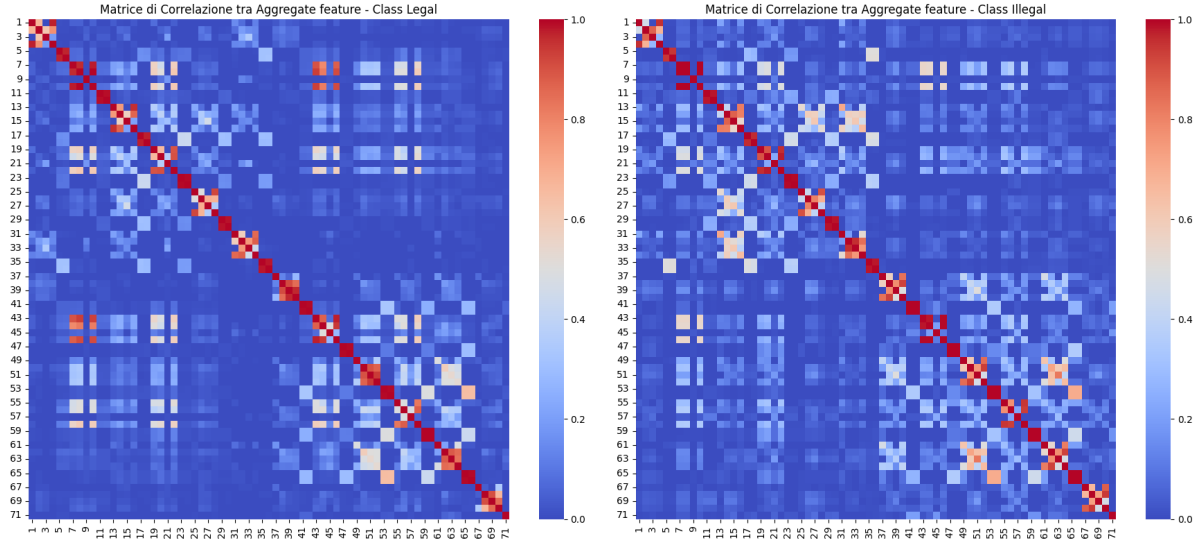

Fig. 1.4: Matrice di Correlazione tra Aggregate feature di tx sulle classi illegal (1) e legal (2)

Un ulteriore approfondimento ha permesso di identificare gruppi di variabili con correlazioni più elevate tra loro (fig. 1.5). In particolare, le variabili relative alle transazioni in Bitcoin mostrano correlazioni interne significative e una stretta relazione con la variabile *total src*. Le correlazioni più evidenti si riscontrano tra le variabili che descrivono i Bitcoin in entrata e in uscita (*in BTC*, *out BTC*), riflettendo una relazione coerente tra gli importi nelle fasi di ingresso e uscita delle transazioni. La variabile temporale *time step* mostra invece una correlazione debole con le altre feature, suggerendo che il momento della transazione non rappresenta un fattore determinante per la struttura del dataset.

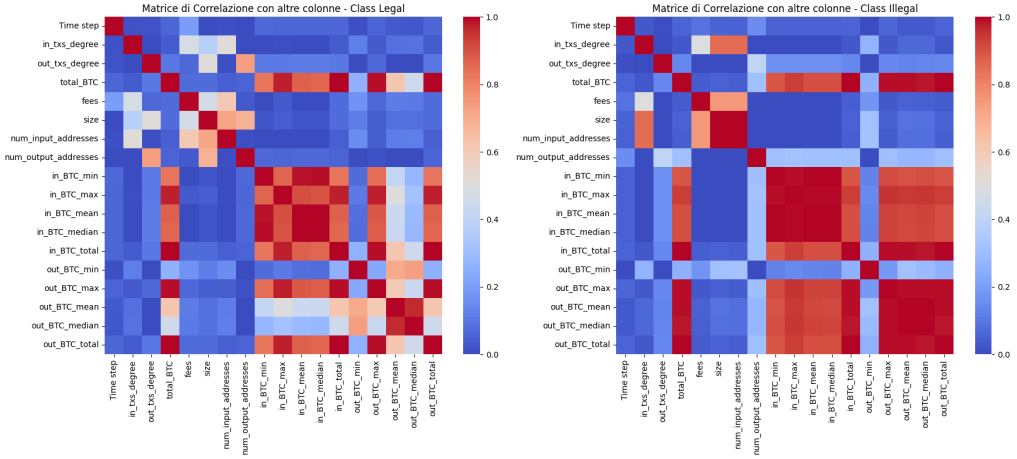

Fig. 1.5: Matrice di Correlazione tra le feature di tx sulle classi illegal (1) e legal (2)

Un aspetto critico emerso durante l'analisi è il forte sbilanciamento della classe target (*class*), figura 1.14.

In particolare, la maggior parte delle istanze appartiene alla classe *unknown* (classe 3), mentre le classi rimanenti, che rappresentano le transazioni di interesse per la classificazione, risultano significativamente sottorappresentate, figura 1.14. Questo squilibrio persiste anche all'interno dei differenti *time step*, mostrando una prevalenza notevolmente superiore di dati sconosciuti, figura 1.7.

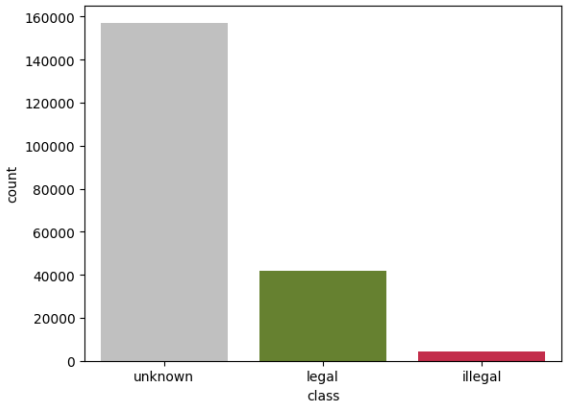

Fig. 1.6: Distribuzione di *class* nel dataset tx

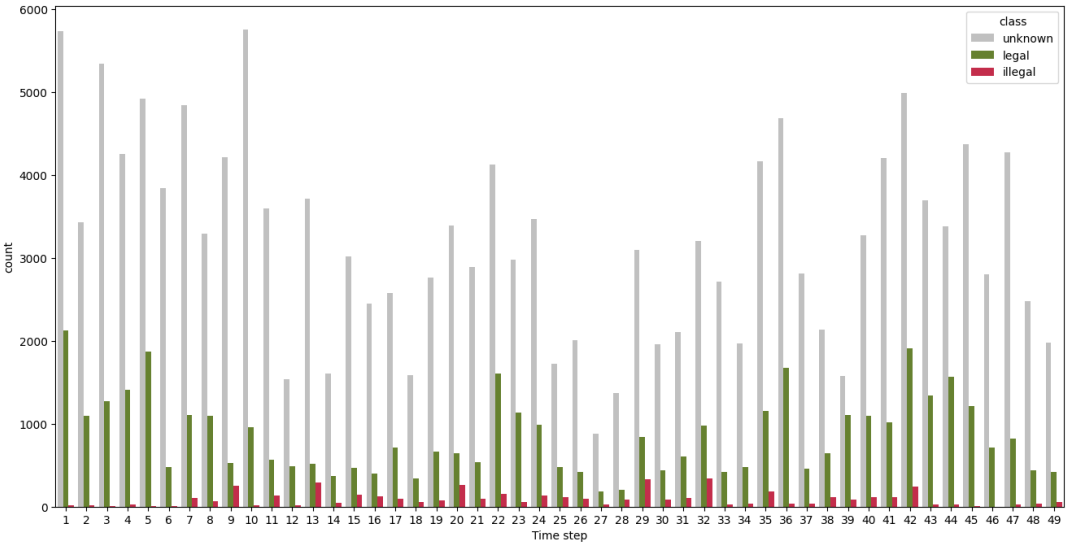

Fig. 1.7: Distribuzione di *class* nel dataset tx suddivisa per times step

Anche eliminando queste istanze, rimane uno sbilanciamento tra le classi lecite e illecite, il che introduce una problematica rilevante nell'addestramento dei modelli predittivi. In effetti, un classificatore potrebbe tendere a favorire la classe maggioritaria, compromettendo la capacità del modello di riconoscere correttamente le transazioni appartenenti alle altre classi.

Social Network Analysis

Il grafo delle transazioni di Elliptic++, composto da 203.769 nodi e 234.355 archi, presenta una densità estremamente bassa (0.000011), riflettendo la natura fortemente sparsa e decentralizzata della rete Bitcoin. Questo valore, unito al criterio di connessione temporale (due transazioni sono collegate solo se avvengono entro tre ore l'una dall'altra), suggerisce che il grafo non rappresenta un sistema completamente interconnesso, ma piuttosto una serie di cluster di interazioni rapide nel tempo. La maggior parte delle transazioni coinvolge un numero limitato di nodi, mentre solo una piccola frazione di essi assume un ruolo centrale nel flusso di Bitcoin. L'analisi della distribuzione dei gradi conferma questa struttura: il grafo segue un modello scale-free, in cui la maggior par-

te dei nodi ha un basso grado, mentre pochi nodi altamente connessi fungono da hub transazionali, facilitando il passaggio di fondi in finestre temporali ristrette.

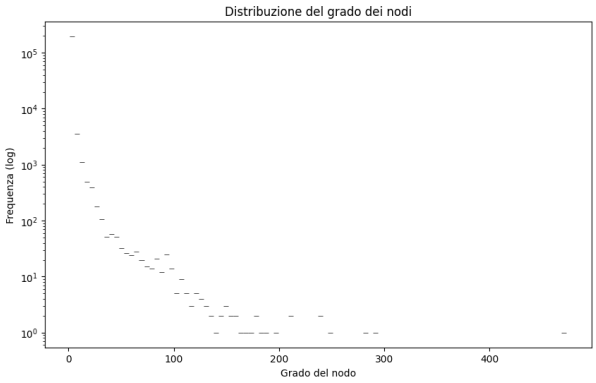

Fig. 1.8: Distribuzione del degree dei nodi di tx

L'average clustering coefficient risulta molto basso (0.01376), suggerendo una scarsa tendenza alla formazione di comunità locali. Questo indica che le transazioni non avvengono in gruppi coesi, ma si distribuiscono in maniera più dispersa, con connessioni lineari piuttosto che strutture fortemente interconnesse. La limitazione temporale delle tre ore rafforza questo comportamento: poiché le transazioni sono collegate solo se avvengono in una finestra temporale ristretta, il clustering rimane basso, riflettendo il carattere dinamico della rete Bitcoin.

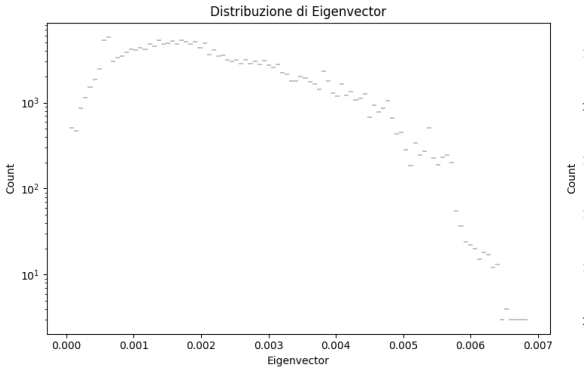

Fig. 1.9: Distribuzione del eigvector dei nodi di tx

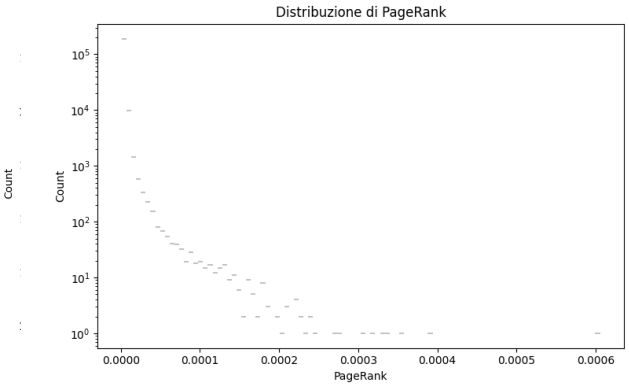

Fig. 1.10: Distribuzione del pagerank dei nodi di tx

L'analisi delle metriche di centralità rafforza questa interpretazione. La closeness centrality ha una media prossima allo zero, con una varianza molto elevata, il che indica che la maggior parte dei nodi è poco raggiungibile rispetto al resto della rete. Solo alcuni nodi si trovano in posizioni strategiche che permettono loro di interagire più rapidamente con il resto del grafo. La eigenvector centrality mostra valori generalmente bassi, suggerendo che solo pochi nodi sono direttamente connessi a transazioni altamente influenti. Anche la PageRank centrality riflette questa struttura, con valori medi molto contenuti e una distribuzione fortemente sbilanciata: ciò significa che la rete è dominata da una piccola frazione di nodi altamente rilevanti, che concentrano il flusso di valore in brevi intervalli temporali.

	mean	std	min	50%	max
Degree	0.000011	0.000021	4.907542e-06	9.815084e-06	0.002321
Closeness	0.000085	0.002214	2.180014e-44	7.185132e-18	0.705157
Eigenvector	0.002126	0.001195	4.626501e-05	1.907587e-03	0.006844
PageRank	0.000005	0.000006	1.119115e-06	4.571930e-06	0.000606

Tab. 1.3: Metriche di centralità

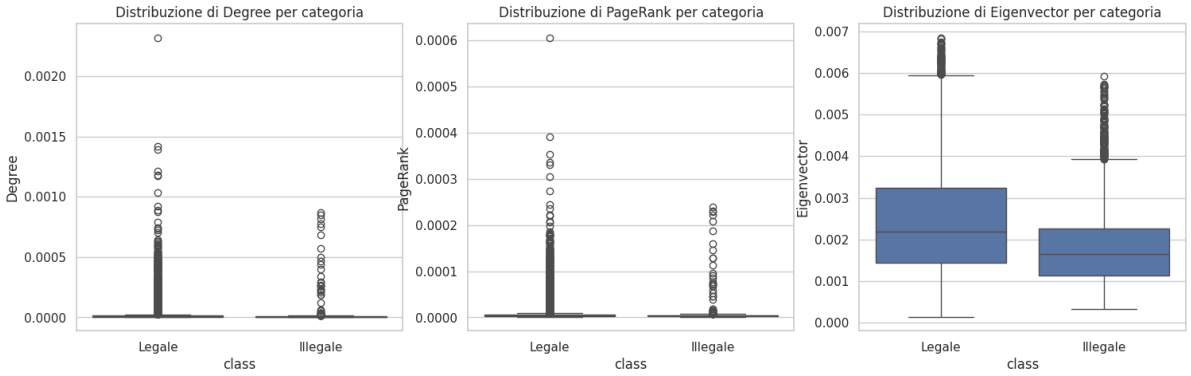

Fig. 1.11: Distribuzione di Degree, PageRank ed Eigenvector Centrality per *class*

La Figura 1.11 mostra la distribuzione delle metriche di centralità (Degree, PageRank ed Eigenvector Centrality) suddivise per categoria (*Legale* e *Illegale*). Dall'analisi emerge che, per tutte le metriche considerate, la distribuzione è fortemente sbilanciata con la presenza di numerosi outlier. Tuttavia, le distribuzioni tra nodi legali e illegali non presentano differenze significative. Per quanto riguarda l'Eigenvector Centrality, si osserva che la mediana è più bassa per i nodi illegali rispetto a quelli legali, ma anche in questo caso le distribuzioni non si discostano in modo rilevante. Questo suggerisce che, sebbene esistano alcune differenze strutturali tra le due categorie, la loro influenza sulla rete rimane relativamente simile.
Nel complesso, questi risultati evidenziano una rete transazionale decentralizzata e fortemente eterogenea, in cui la maggior parte delle transazioni avviene attraverso nodi periferici con bassa connettività.

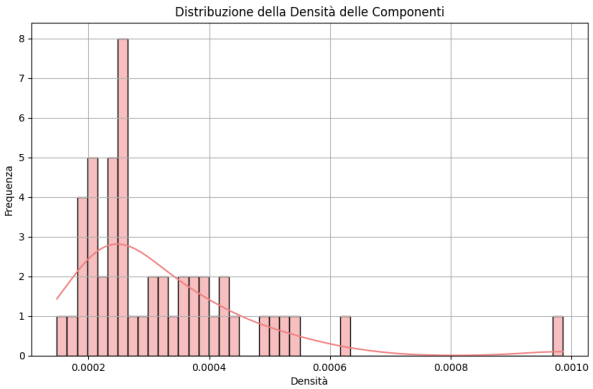

Fig. 1.12: Distribuzione della densità delle componenti di tx

Il grafo è composto da 49 componenti connesse, indicando che la rete non è un sistema monolitico, ma piuttosto un insieme di sottoreti di breve durata. La componente più grande include 7.880 nodi, confermando che la maggior parte delle transazioni avviene in cluster più piccoli e indipendenti rispetto al grafo originario, inoltre la loro densità oscilla tra 0.00014 e 0.00098, riflettendo nuovamente una struttura sparsa anche nei sottografi.

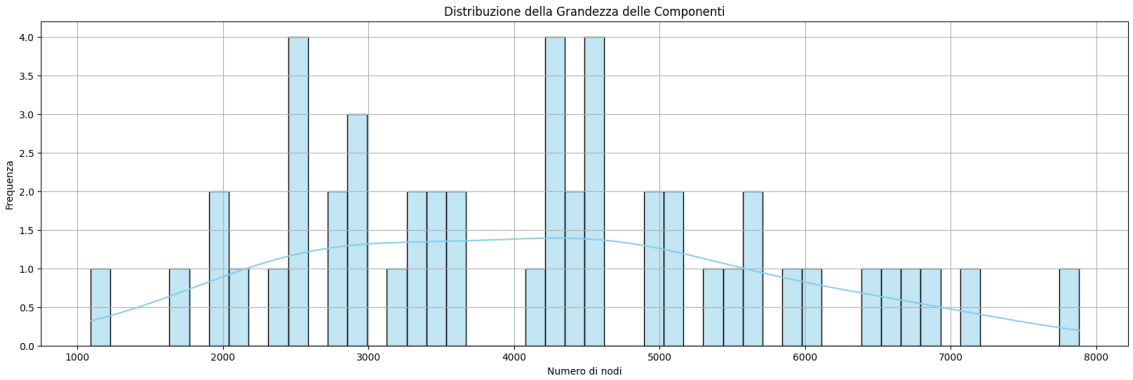

Fig. 1.13: Distribuzione della dimensione dei nodi di tx

La componente connessa più grande è costituita da 7.880 nodi e 9.164 archi, ma rimane comunque di dimensioni ridotte rispetto all'intero grafo. Presenta una bassa densità (0.000148) e un grado medio di 2.33, con un valore massimo di 289. Il coefficiente di clustering medio (0.008) è molto basso, suggerendo che la struttura della rete non è altamente coesa e che le connessioni tra i nodi tendono a formare strutture lineari piuttosto che comunità dense.
Infine, la componente più piccola tra le 49 principali conta 1.089 nodi e 1.168 archi, con una densità di 0.000986. Il suo coefficiente di clustering medio (0.021) è leggermente più alto rispetto alle altre componenti, suggerendo che qui le transazioni tendono a formare piccoli gruppi con maggiore interconnessione locale.

Il grado medio dei nodi nelle diverse componenti si attesta tra 2.05 e 2.87, indicando che la maggior parte dei nodi possiede pochissime connessioni. Tuttavia, il grado massimo varia significativamente (da 37 a 473), suggerendo la presenza di hub transazionali attivi in brevi finestre temporali.

	Componente 1	**Componente 49**
component_size	7880	1089
num_edges	9164	1168
density	0.000148	0.000986
avg_degree	2.325888	2.145087
max_degree	289	95
avg_clustering	0.008084	0.021325

Tab. 1.4: Confronto tra le componenti

1.2.2 Dataset dei Wallet

Analisi Statistica

Dopo un'analisi preliminare sulle transazioni, si è approfondito lo studio dei wallet con l'obiettivo di individuare eventuali pattern e correlazioni tra le variabili, nonché comprendere meglio la struttura del dataset. Questo processo è stato inoltre fondamentale per determinare le strategie più efficaci in fase di addestramento del modello.

L'analisi ha permesso di distinguere tre macro-categorie principali di informazioni contenute nel dataset:

- **Generali**: informazioni identificative degli indirizzi e della classe target, che assume i valori 1 (*illegal*), 2 (*legal*) e 3 (*unknown*).
- **Transaction-related**: metriche relative alle transazioni effettuate, con particolare attenzione agli importi in Bitcoin (BTC) e alle commissioni. Un sottogruppo di queste metriche descrive le interazioni tra diversi indirizzi Bitcoin.
- **Time-related**: metriche temporali legate ai blocchi e alla loro distribuzione, con un sottogruppo specifico dedicato agli intervalli tra transazioni successive.

Le colonne del dataset possono essere suddivise nel seguente modo:

Categoria	**Colonne**
Generali	*address*, *class*
Transaction-related	*num_txs_as_sender*, *num_txs_as_receiver*, *total_txs*, *btc_transacted_**, *btc_sent_**, *btc_received_**, *fees_**, *fees_as_share_**
(Interazioni)	*num_addr_transacted_multiple*, *transacted_w_address_**
Time-related	*Time step*, *num_timesteps_appeared_in*, *lifetime_in_blocks* *first_block_appeared_in*, *last_block_appeared_in*, *first_sent_block*, *first_received_block*
(Intervalli)	*blocks_btwn_txs_**, *blocks_btwn_input_txs_**, *blocks_btwn_output_txs_**,

Tab. 1.5: Macro-categorie e sottoinsiemi delle colonne di wallet

Sono presenti colonne (contrassegnate con * nella tabella 1.5) che contengono valori statistici relativi alla loro categoria, tra cui media, mediana, massimo, minimo e totale, fornendo così una visione più completa dei dati a disposizione.

Come riscontrato in tx, figura 1.14, anche nei wallet si riscontra una distribuzione fortemente sbilanciata dei valori della classe target, che emerge sia analizzando i dati nella loro interezza, sia eseguendo un'analisi sui distinti time step. In particolare il 71% dei wallet presenti nel grafo appartiene alla classe unknown, mentre il 26,7% sono classificati come legal e solo il rimante 2,3% è appratente alla classe **illegal**.

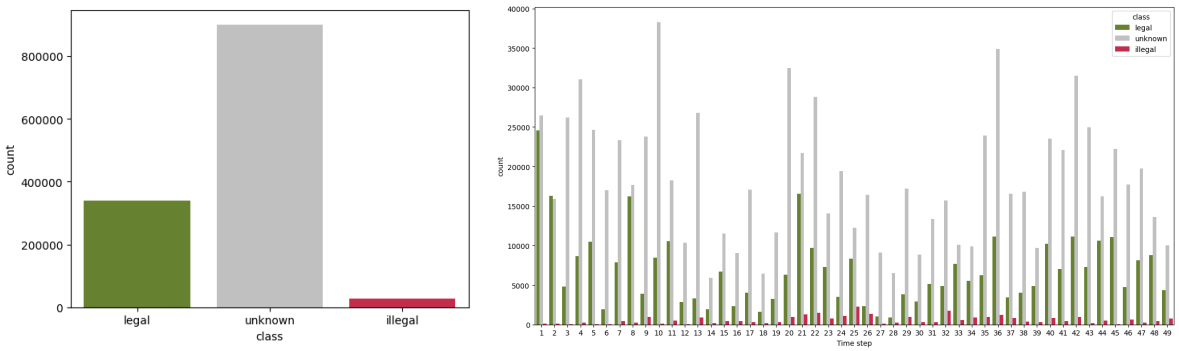

Fig. 1.14: Distribuzione di *class* nel dataset wallet, sull'intero dataset e suddivisa per singoli intervalli temporali

Di seguito le analisi effettuate saranno mirate alla comprensione del dataset per la classificazione binaria. Verranno, pertanto, esclusi gli elementi con valore 'unknown', mantenendo solo le classi *legal* e *illegal*, che saranno analizzate separatamente al fine di individuare meglio i pattern e le caratteristiche che le contraddistinguono.

Analisi delle metriche temporali Una prima analisi verteva ad approfondire lo studio delle colonne principali della categoria *Time-related*, in particolare:

- **Timesteps**: indica il numero di periodi temporali distinti in cui un indirizzo ha effettuato transazioni.
- **Lifetime**: rappresenta il numero totale di blocchi durante i quali un indirizzo è rimasto attivo.

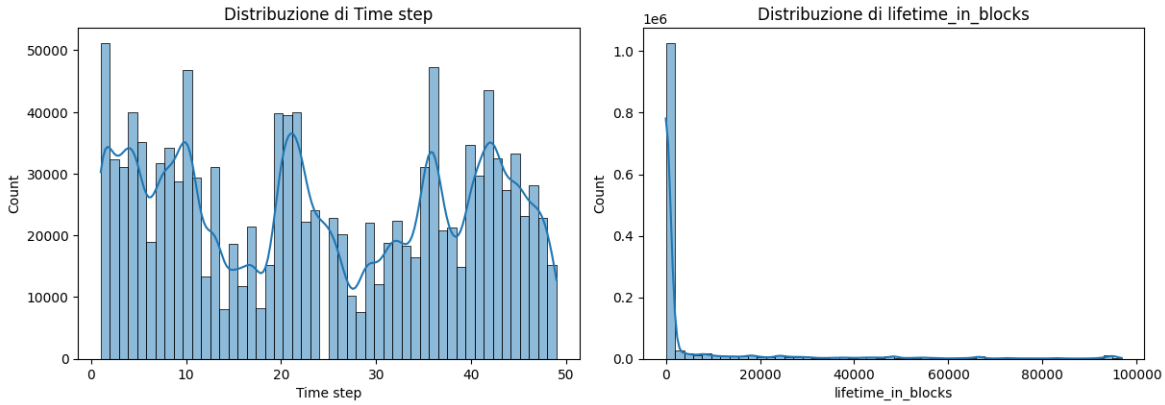

Fig. 1.15: Distribuzione delle colonne *Time steps* e *Lifetime in Blocks*

Come mostrato nella figura 1.15, la distribuzione di *Time steps* è piuttosto distribuita, con alcune variazioni periodiche. Tuttavia, un dato più significativo emerge dall'analisi della colonna *Lifetime in Blocks*: questa presenta una forte asimmetria, con un'alta concentrazione di indirizzi caratterizzati da una vita breve e solo pochi con una durata molto lunga. Questo risultato suggerisce che la maggior parte degli indirizzi esegua transazioni in un arco temporale limitato.

Successivamente, si è indagata la correlazione tra le metriche temporali e le altre variabili del dataset, al fine di individuare eventuali pattern significativi. Per questa analisi è stato utilizzato l'indice di correlazione di Pearson, calcolato separatamente per le classi *legal* e *illegal*..

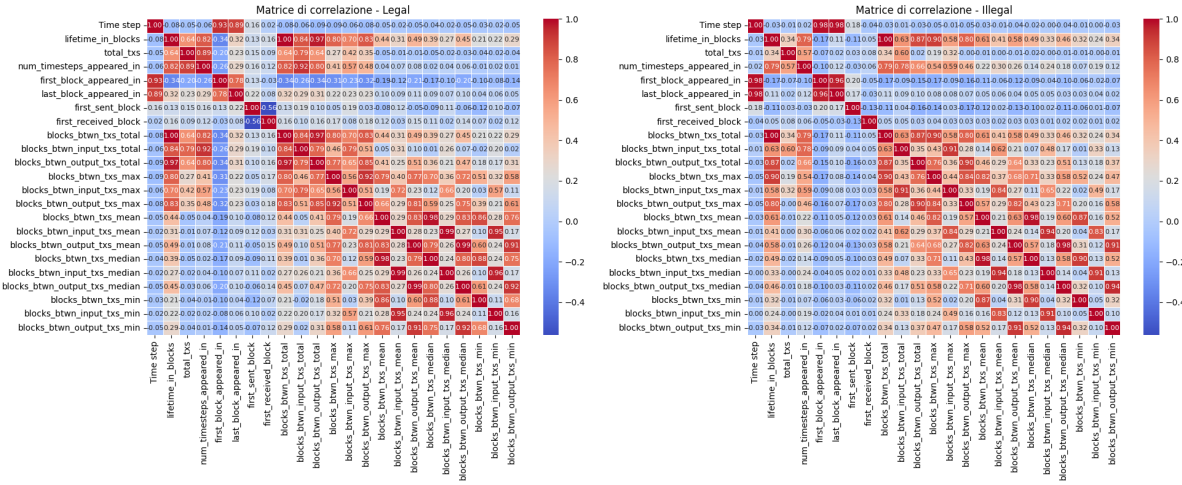

Fig. 1.16: Matrice di Correlazione tra le feature time related di wallet sulle classi illegal (1) e legal (2)

Contrariamente alle aspettative, le correlazioni tra le variabili risultano essere molto simili tra le istanze della classe *legal* e quelle della classe *illegal*. In entrambe le matrici si osserva un sottogruppo di colonne fortemente correlate, appartenenti alla sottocategoria degli intervalli tra transazioni. Questa correlazione può essere attribuita sia alla natura intrinseca dei dati sia al fatto che molte colonne rappresentano statistiche descrittive delle stesse informazioni (ad esempio, valori massimi, minimi e medi).
Un altro risultato interessante riguarda la colonna `lifetime_in_blocks`, che mostra una correlazione significativa con quasi tutte le altre variabili della macrocategoria *Time-related*. Fanno eccezione alcune colonne come `time_step`, `first_block`, `last_block`, `first_send_block` e `first_received_block`, che invece risultano maggiormente correlate con `time_step`.
Un'ulteriore osservazione riguarda la presenza di pattern ricorrenti in alcune colonne, in particolare quelle relative agli intervalli tra transazioni presentano valori che tendono a seguire schemi ripetitivi, sebbene non così marcate come quelle osservate nelle metriche di transazione (figura 1.4 e 1.1). La presenza di molte colonne correlate suggerisce che l'applicazione di tecniche di riduzione della dimensionalità potrebbe portare benefici in termini di performance del modello, infatti tecniche come la PCA consentirebbero di catturare le informazioni più rilevanti minimizzando la ridondanza e migliorando la generalizzazione del modello predittivo.
Per completare l'analisi, è stata inclusa anche la colonna `total_txs` (originariamente appartenente alla macrocategoria *Transaction-related*) al fine di indagare se il numero totale di transazioni fosse correlato con le metriche temporali. L'ipotesi iniziale era che gli indirizzi classificati come *illegal* effettuassero un numero maggiore di transazioni in periodi di tempo limitati.

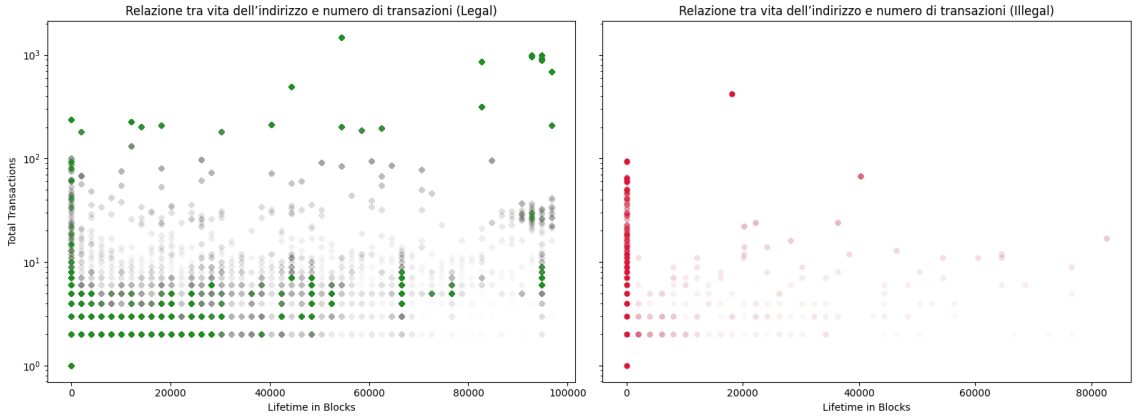

Fig. 1.17: Analisi della distribuzione tra *total_txs* e *lifetime_in_blocks* nelle diverse classi

Dall'analisi emerge, come mostrato nei grafici della figura 1.17, che i wallet illeciti si distribuiscono con una maggiore concentrazione attorno a lifetime prossimi allo zero, supportando l'ipotesi che le istanze della classe *illegal* presentano transazioni in brevi intervalli di tempo, ma smentendo l'idea che questi indirizzi eseguano un numero maggiore di transazioni rispetto alla classe *legal*. Inoltre, va evidenziato che anche la classe *legal*, pur mostrando una distribuzione più omogenea delle istanze rispetto a lifetime, presenta una concentrazione più elevata per valori prossimi allo zero. Queste considerazioni sono coerenti anche con l'analisi eseguita sulla distribuzione di lifetime (figura 1.15) che vede una distribuzione fortemente sbilanciata verso lo 0.

Questo pattern, sebbene interessante, non è sufficiente a distinguere in modo chiaro le due classi, soprattutto tenendo in considerazione la differenza nel numero di istanze tra la classe *legal* e quella *illegal* analizzate.

L'analisi iniziale condotta non ha evidenziato pattern o correlazioni significative che consentissero di distinguere i wallet classificati come *legal* e *illegal*, per questo motivo, si è proceduto con un'analisi più approfondita al fine di individuare possibili wallet sospetti, ovvero indirizzi che mostrano comportamenti anomali nelle transazioni.
Una prima verifica è stata condotta per individuare la presenza di eventuali bot: inizialmente, sono stati analizzati gli indirizzi con un'elevata frequenza di transazioni, ma non è stato possibile identificare wallet con caratteristiche chiaramente sospette. Successivamente, l'analisi si è focalizzata sugli indirizzi che presentano transazioni a intervalli regolari, ipotizzando che tale comportamento potesse indicare un'attività automatizzata. È emerso che il 42.3% degli indirizzi presenti nel dataset (155450 indirizzi) mostra questa caratteristica; tuttavia, solo il 15.35% degli indirizzi appartenenti alla classe *illegal* (rispetto al totale degli indirizzi di questa classe del dataset) presenta questo pattern, suggerendo che tale comportamento non sia un forte indicatore di attività illecite.
Una seconda verifica ha riguardato l'analisi del mixing dei fondi, tecnica utilizzata per offuscare la provenienza dei Bitcoin rendendo più difficile tracciarne l'origine; questa tecnica è spesso impiegata per aumentare la privacy delle transazioni, ma può anche essere utilizzata per mascherare attività illecite. In particolare, sono stati esaminati gli indirizzi che effettuano transazioni all'interno dello stesso blocco (`blocks_btwn_txs_min` = 0), poiché l'esecuzione simultanea di transazioni potrebbe rappresentare un tentativo di confondere l'origine e la destinazione dei fondi, rendendo più complesso il tracciamento. È stato rilevato che il 71.94% degli indirizzi totali (264350 indirizzi) presenta questa caratteristica; tuttavia, solo il 4.16% degli indirizzi *illegal* ricade in questa categoria, suggerendo che il mixing, nel contesto del dataset analizzato, non sia un forte predittore di attività illecite.
Un'ulteriore analisi è stata condotta per identificare eventuali pattern di accumulo di BTC, ossia situazioni in cui un indirizzo riceve BTC con lunghi intervalli temporali ma li spende rapidamente, ipotizzando che tale comportamento possa indicare un'attività di raccolta prima di un'operazione di cashout. Per verificare questa ipotesi, sono stati confrontati i tempi delle transazioni in ingresso e in uscita. L'analisi ha evidenziato che solo l'1.47% degli indirizzi totali (5421 indirizzi) mostra questo comportamento, con un'incidenza trascurabile (0.33%) tra gli indirizzi *illegal* (rispetto al totale degli indirizzi *illegal* nel dataset), suggerendo che l'accumulo di BTC non sia un forte indicatore di attività illecite. Nel complesso, queste analisi suggeriscono che, sebbene alcuni pattern anomali possano essere identificati, non esiste una caratteristica univoca che permetta di distinguere con certezza gli indirizzi *illegal* da quelli *legal*.

Analisi delle metriche delle transazioni L'analisi preliminare delle correlazioni tra le variabili non ha evidenziato la presenza di colonne correlata con la maggior parte delle altre; al contrario, si osservano correlazioni a blocchi, come già emerso in precedenti analisi su altre variabili. Tuttavia, rispetto a tali analisi precedenti, le correlazioni a blocchi risultano meno nette, in particolare, nella classe legal, la distinzione tra i blocchi di correlazione appare quasi trascurabile, come illustrato nella Figura 1.18.

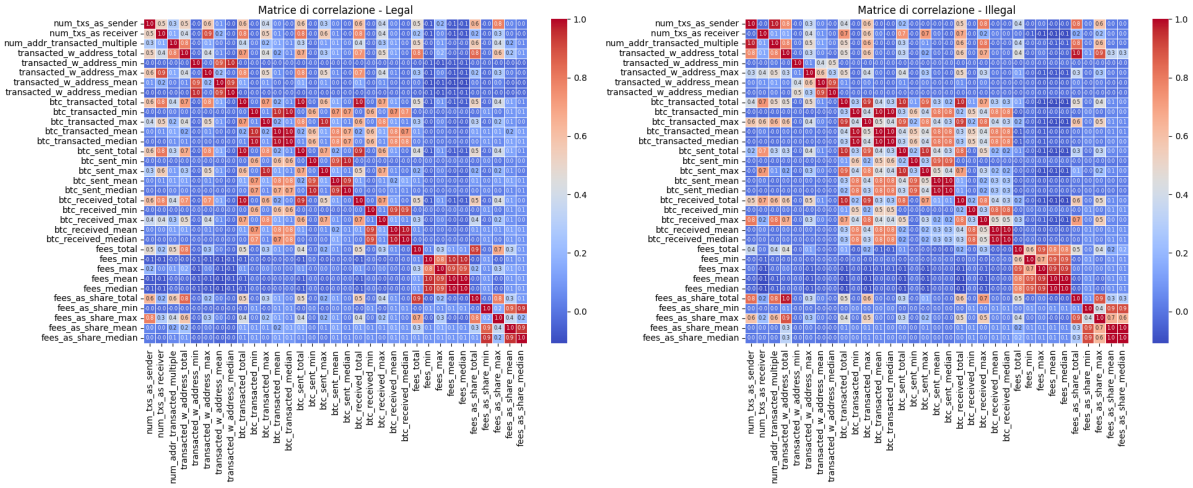

Fig. 1.18: Matrice di Correlazione tra le feature transaction related di wallet sulle classi illegal (1) e legal (2)

Nonostante l'assenza di correlazioni marcate, è possibile individuare due principali sottogruppi di variabili:

- Variabili relative alle transazioni in Bitcoin (BTC): che mostrano principalmente misure descrittive relative alle transazioni in bitcoin
- Variabili relative alle commissioni di transazione (fees): in cui si osserva una suddivisione tra le correlazioni relative alle commissioni totali e quelle fees_share.

Analogamente a quanto già osservato nell'analisi delle variabili temporali, non emergono pattern di correlazione distintivi che consentano di discriminare in modo netto tra le classi legal e illegal. Inoltre anche in questo caso la struttura delle correlazioni suggerisce che l'applicazione di tecniche di riduzione della dimensionalità potrebbe fornire, sebbene in misura limitata, miglioramenti in termini di efficienza computazionale e performance del modello.

Sono stata analizzate in modo più approfondito le relazioni tra il numero di transazioni inviate, ricevute e il numero di indirizzi che hanno interagito più volte. Questo approfondimento è motivato dall'osservazione, illustrata nella figura 1.18, di correlazioni differenti tra queste variabili nelle classi legal e illegal.

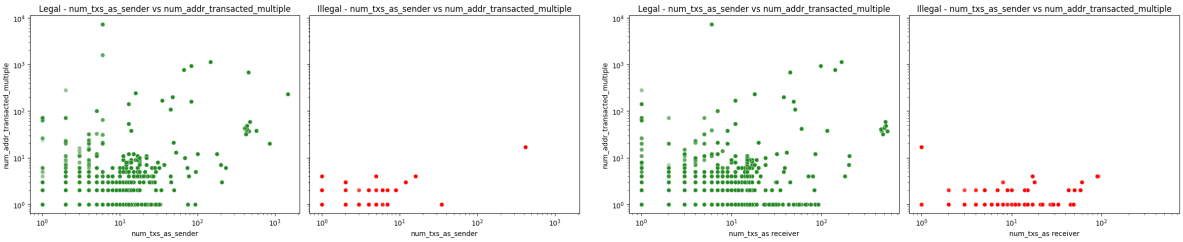

Fig. 1.19: Relazioni tra numero di transazioni inviate, ricevute e di indirizzi che hanno interagito più volte per *class*

L'analisi ha riscontrato effettivamente differenze tra le due classi: le transazioni appartenenti alla classe illegal presentano in genere un numero inferiore sia di indirizzi con più interazioni (num_addr_transacted_multiple) sia di transazioni eseguite in qualità di mittente (num_tx_as_sender). Mentre per quanto riguarda le transazioni ricevute, si osserva che i valori di num_addr_transacted_multiple risultano concentrati su valori bassi per la classe illegal, mentre mostrano una maggiore dispersione nella classe legal. Tuttavia, la distribuzione del numero di transazioni ricevute (num_tx_as_receiver) risulta simile per

entrambe le classi.
Poiché le distribuzioni sembrano comparabili, si è proceduto ad analizzare il rapporto tra il numero di transazioni inviate e ricevute per ciascuna classe.

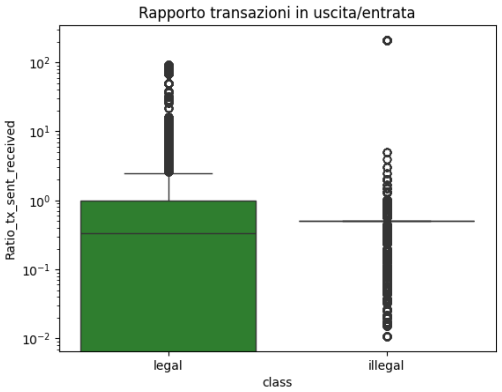

Fig. 1.20: Distribuzione del rapporto tra transazioni in uscita ed entrata

L'analisi suggerisce che gli indirizzi *illeciti* tendono a mantenere un rapporto più bilanciato tra transazioni in entrata e in uscita, come evidenziato nel boxplot 1.20, con una maggiore concentrazione attorno al valore 1.
Al contrario, gli indirizzi legittimi mostrano un rapporto maggiore, indicando una tendenza a inviare più transazioni di quante ne ricevano.

Questo dato potrebbe essere coerente con un comportamento comune nel riciclaggio di denaro che cerca di mantenere un bilanciamento tra transazioni in entrata e in uscita per evitare di destare sospetti.
Tale differenza potrebbe essere sfruttata per costruire caratteristiche ingegnerizzate utili alla classificazione degli indirizzi.

In conclusione, nonostante le analisi condotte, non emergono correlazioni o pattern significativi per l'identificazione univoca delle istanze *illecite*. L'assenza di segnali distintivi suggerisce che le metriche considerate, almeno singolarmente, non siano sufficienti per distinguere chiaramente le due classi, richiedendo l'integrazione di ulteriori caratteristiche o l'adozione di approcci modellistici più avanzati.

Social Network Analysis

Il grafo dei wallet è composto da 2.293.542 nodi e 4.049.441 archi, con una densità pari a 0.000002. Questa densità estremamente bassa indica una rete altamente decentralizzata e disomogenea, caratterizzata da poche connessioni tra i wallet. La maggior parte dei nodi ha un numero limitato di connessioni, mentre solo una piccola frazione svolge il ruolo di hub.

Per condurre un'analisi efficace delle comunità e calcolare le principali metriche di centralità (Degree, PageRank ed Eigenvector Centrality), è stato necessario un preprocessing del grafo per ottimizzare l'uso delle risorse computazionali. In particolare, è stata creata una copia semplificata del grafo contenente esclusivamente gli ID dei nodi. Questa operazione si è resa indispensabile a causa delle dimensioni del dataset, permettendo una riduzione del carico computazionale e garantendo un'elaborazione più efficiente. Per l'analisi della struttura della rete, sono state calcolate tre metriche di centralità utilizzando NetworKit, una libreria ottimizzata per grandi grafi.

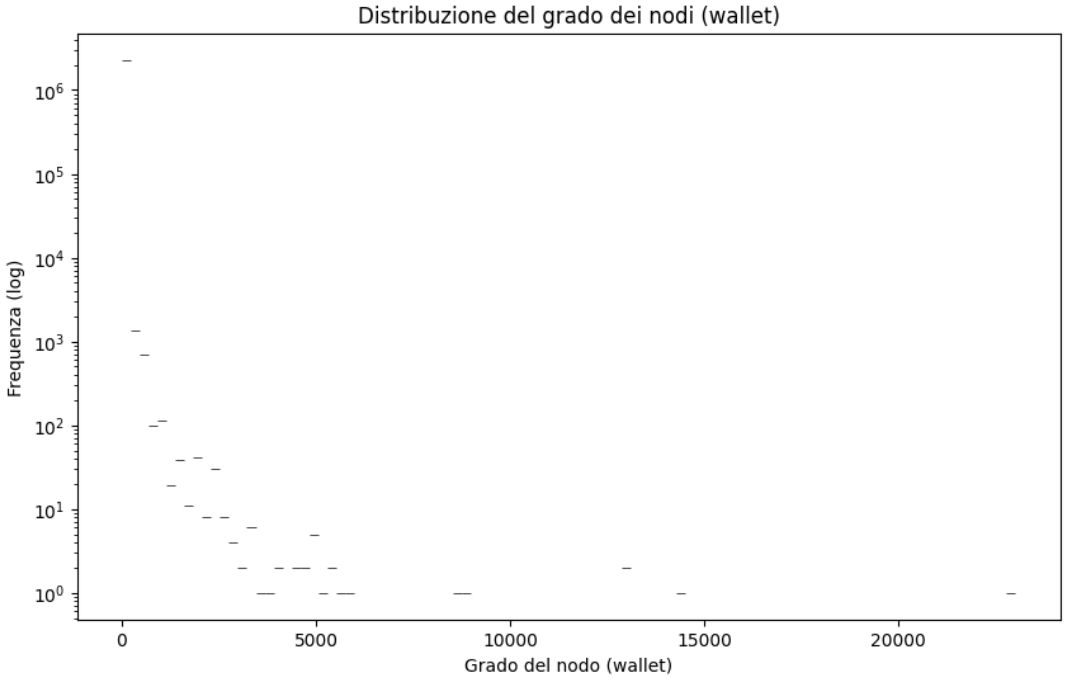

Fig. 1.21: Distribuzione del degree dei nodi di wallet

La *Degree Centrality* evidenzia che la maggior parte dei nodi ha un valore molto basso, indicando che la rete è composta principalmente da wallet con poche connessioni. Tuttavia, una piccola frazione di nodi presenta un grado significativamente più elevato, suggerendo la presenza di wallet che fungono da intermediari o punti di aggregazione. Questa distribuzione segue una legge di potenza (power-law), tipica delle reti scale-free. Analogamente, la *Eigenvector Centrality* riflette questa struttura, evidenziando che solo un numero limitato di nodi è connesso a punti altamente centrali, mentre la maggior parte dei wallet ha un ruolo marginale. Il *PageRank* conferma questa tendenza, indicando che l'influenza sulla rete è concentrata in pochi nodi chiave, mentre la maggioranza dei wallet ha un impatto trascurabile sulla propagazione delle transazioni.

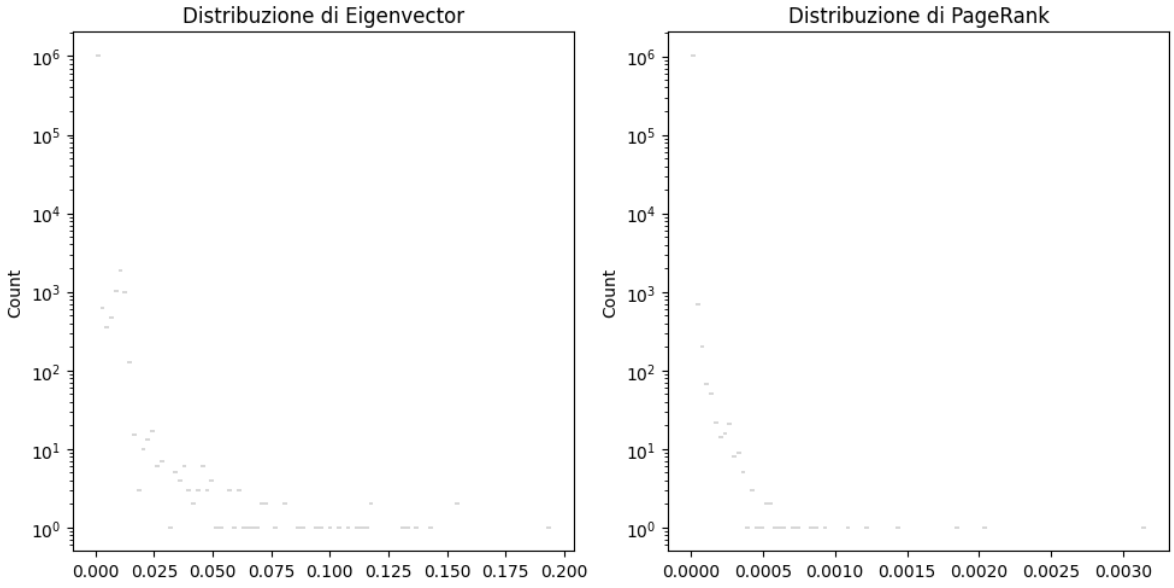

Fig. 1.22: Distribuzione del eigenvector e pagerank dei nodi di wallet

	mean	std	min	50%	max
Degree	0.000008	0.000052	0.000001	0.000003	0.022431
Eigenvector	0.000062	0.000985	0.000000	0.000000	0.194356
PageRank	0.000001	0.000006	0.000000	0.000001	0.003159

Tab. 1.6: Metriche di centralità

Un'analisi più approfondita delle metriche tra wallet legali (classe 2) e illegali (classe 1) mostra che questa struttura si mantiene indipendentemente dalla natura dei nodi 1.23. Non emergono differenze significative tra le due categorie: i nodi illegali non presentano valori anomali rispetto alla distribuzione complessiva della rete, e la presenza di hub con grado elevato è comune a entrambe le classi. Questo suggerisce che i wallet coinvolti in attività illecite siano strutturalmente simili a quelli legali in termini di connettività. L'assenza di differenze marcate nelle metriche di centralità indica che le transazioni illecite sono perfettamente integrate nel flusso generale della rete, rendendo più complessa la loro individuazione attraverso sole analisi strutturali.

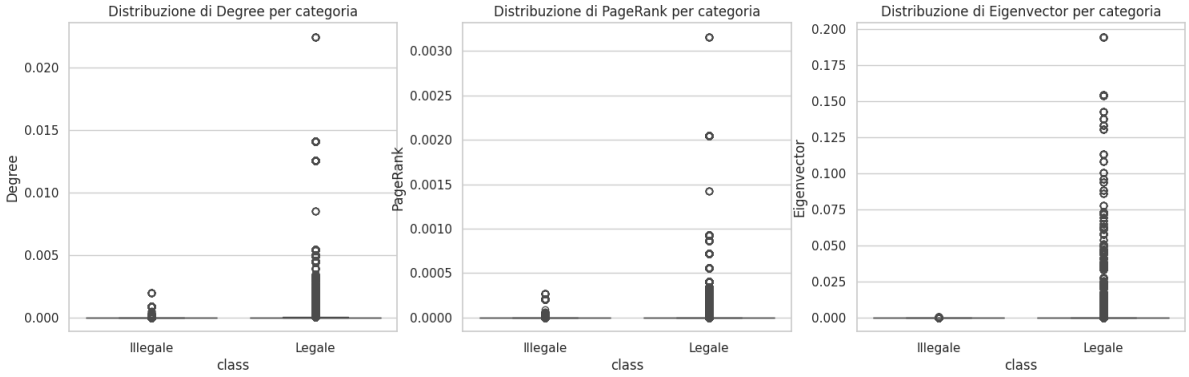

Fig. 1.23: Distribuzione di Degree, PageRank ed Eigenvector Centrality per categoria

Per approfondire ulteriormente la relazione tra influenza nella rete e attività transazionali, è stata analizzata la connessione tra il *PageRank* e il volume totale di Bitcoin transati. I risultati mostrano che i nodi con un *PageRank* più elevato tendono a gestire volumi maggiori di BTC, suggerendo una correlazione tra centralità e quantità di Bitcoin scambiati 1.24. Anche in questo caso, i nodi illegali risultano distribuiti in modo uniforme all'interno della rete, senza anomalie rispetto ai nodi legali. Questo implica che i wallet coinvolti in attività illecite non si distinguono per volumi di transazione atipici e che alcuni di essi possiedono sia un *PageRank* elevato sia volumi significativi di BTC, confermando il loro ruolo chiave nella rete transazionale. L'integrazione dei wallet illeciti nella distribuzione complessiva suggerisce quindi l'assenza di pattern strutturali evidenti che li separino dai nodi legali, rendendo necessarie analisi più approfondite per la loro identificazione.

Parallelamente, è stata condotta un'analisi specifica per verificare l'esistenza di archi diretti tra wallet illegali. I risultati mostrano che non esistono coppie di nodi illegali direttamente collegate tra loro, ad eccezione di una singola coppia con distanza uno. Questo significa che, nella quasi totalità dei casi, i wallet illegali non interagiscono direttamente tra loro, ma si distribuiscono nella rete attraverso connessioni intermedie, rafforzando ulteriormente l'idea che i wallet illegali siano perfettamente integrati nella rete complessiva, senza una struttura isolata o facilmente identificabile.

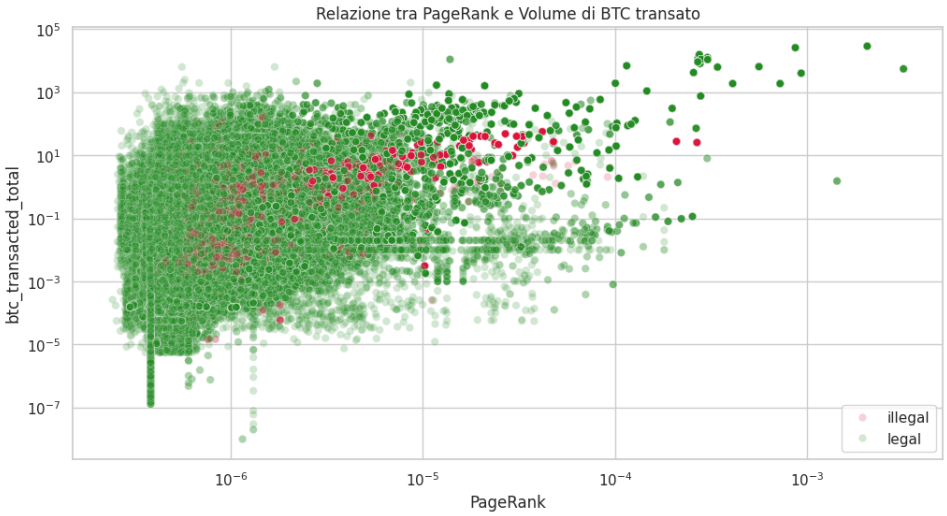

Fig. 1.24: Relazione tra PageRank e volume di BTC transato, colorato per classe

Questa natura frammentata della rete viene riconfermata anche dall'analisi delle componenti connesse. Il grafo è infatti suddiviso in 1.267.798 componenti connesse, il che significa che la maggior parte dei nodi non è direttamente collegata tra loro, ma esistono numerosi sotto-gruppi o cluster di wallet che interagiscono solo al loro interno. Inoltre, è interessante notare che la componente connessa più grande contiene 1.025.738 nodi, quasi la metà dell'intero grafo, e 4.049.422 archi rispetto ai 4.049.441 archi totali. Questo suggerisce che la maggior parte delle altre componenti siano in realtà nodi isolati o piccoli gruppi con pochissime connessioni. In seguito a questa analisi, è stato verificato a quale classe appartengano i nodi esclusi dalla componente connessa più grande. È emerso che tutti questi nodi sono stati etichettati come "unknown", evidenziando che la loro attività sia di interesse marginale.
L'analisi della rete dei wallet nel dataset Elliptic++ evidenzia una struttura decentralizzata e frammentata, con pochi nodi chiave che dominano il flusso delle transazioni. I nodi illegali non mostrano particolari anomalie strutturali, rendendo complessa la loro individuazione tramite le sole metriche di centralità, senza pattern distintivi che ne facilitino il riconoscimento.

Capitolo 2

Modelli

2.1 Modelli

2.1.1 Graph Neural Networks (GNNs)

Le Graph Neural Networks (GNNs) sono una classe di modelli di deep learning progettati per gestire dati strutturati come grafi, dove le relazioni tra entità (nodi) sono rappresentate da archi. Le GNNs si basano su un meccanismo iterativo chiamato message passing, che consente ai nodi di aggiornare le proprie rappresentazioni sfruttando le informazioni provenienti dai nodi vicini. Questo processo si articola principalmente in due fasi: aggregazione (aggregate) e combinazione (combine).

1. **Aggregazione (Aggregate):** Per ogni nodo v, le informazioni provenienti dai nodi adiacenti $\mathcal{N}(v)$ vengono raccolte tramite una funzione di aggregazione invariante rispetto alla permutazione dei nodi, garantendo la coerenza delle rappresentazioni indipendentemente dall'ordinamento dei vicini. La funzione di aggregazione può essere definita formalmente come:
$$m_v^{(k)} = \text{AGG}\left(\left\{h_u^{(k-1)} \mid u \in \mathcal{N}(v)\right\}\right) \text{[8]}$$
dove $h_u^{(k-1)}$ rappresenta l'embedding del nodo u al layer $k-1$, e $m_v^{(k)}$ è il messaggio aggregato per il nodo v al layer k. Funzioni comuni di aggregazione includono somma, media o massimizzazione, ma possono essere utilizzate anche funzioni più complesse basate su attention mechanisms.
2. **Combinazione (Combine):** Successivamente, il nodo v aggiorna la propria rappresentazione combinando il messaggio aggregato $m_v^{(k)}$ con la sua rappresentazione precedente $h_v^{(k-1)}$. Questa fase può essere formalizzata come:
$$h_v^{(k)} = \text{COMBINE}\left(h_v^{(k-1)}, m_v^{(k)}\right) \text{[8]}$$
La funzione di combinazione può essere una semplice concatenazione, una somma pesata o una funzione appresa.

Questo processo viene iterato per K strati, consentendo ai nodi di incorporare informazioni provenienti da vicinati sempre più ampi. Dopo K iterazioni, la rappresentazione finale di ciascun nodo $h_v^{(K)}$ cattura informazioni sia locali che globali del grafo.
Nel caso specifico del grafo di Elliptic++, caratterizzato da transazioni collegate da archi definiti entro un intervallo temporale di 3 ore, l'utilizzo delle GNNs consente di modellare in maniera efficace le dipendenze temporali e topologiche. Questo approccio permette di enfatizzare le connessioni recenti e rilevanti, facilitando l'identificazione di nodi strategici (come exchange o mixer) e il rilevamento di pattern di transazione sospetti.

2.1.2 Architetture: SAGE, GAT e Transformer

Nel modello proposto sono state utilizzate tre diverse tipologie di di GNN per il processo di apprendimento: **SAGEConv**, **Graph Attention Network (GAT)** e **Graph Transformer**, ciascuna caratterizzata da distinti meccanismi di aggregazione e combinazione delle informazioni.

SAGEConv

SageConv [4] è un'estensione di GraphSAGE che introduce un'aggregazione basata sulla media normalizzata rispetto al grado dei nodi vicini, migliorando la capacità del modello di catturare informazioni strutturali del grafo. Inoltre, l'integrazione di connessioni skip facilita il flusso del gradiente, mitigando il problema della scomparsa del gradiente nelle reti profonde e preservando le informazioni dai livelli precedenti. Rispetto ad altre architetture di reti neurali su grafi (GNN), SageConv garantisce una maggiore efficienza computazionale, evitando operazioni onerose come inversioni di matrice o decomposizioni spettrali.

Graph Attention Network (GAT)

GAT [1] introduce meccanismi di attenzione per pesare dinamicamente l'importanza dei nodi vicini. Per ogni coppia di nodi (v, u), viene calcolato un coefficiente di attenzione:

$$\alpha_{vu} = \frac{\exp\left(\text{LeakyReLU}\left(\mathbf{a}^\top [Wh_v \| Wh_u]\right)\right)}{\sum_{k \in \mathcal{N}(v)} \exp\left(\text{LeakyReLU}\left(\mathbf{a}^\top [Wh_v \| Wh_k]\right)\right)} [11]$$

dove $\mathbf{a}$ è un vettore di pesi apprendibili, W è una matrice di trasformazione, e $\|$ indica la concatenazione. La nuova rappresentazione del nodo è data da:

$$h_v^{(k)} = \sigma\left(\sum_{u \in \mathcal{N}(v)} \alpha_{vu} W h_u\right) [11]$$

Questo meccanismo consente al modello di focalizzarsi sui vicini più informativi, migliorando la discriminazione tra transazioni rilevanti e irrilevanti.

Graph Transformer

Il **TransformerConv** di PyG [5] estende il paradigma del message passing nei grafi incorporando un meccanismo di attenzione auto-supervisionata, ispirato all'architettura dei Transformer, come descritto nel lavoro *Masked Label Prediction: Unified Message Passing Model for Semi-Supervised Classification* [10]. Questo approccio consente ai nodi di apprendere dinamicamente l'importanza relativa dei messaggi ricevuti dai vicini, superando la limitazione delle aggregazioni uniformi tipiche dei GCN tradizionali. L'aggiornamento della rappresentazione di un nodo v al livello ℓ per la testa di attenzione c è definito come:

$$h_v^{(\ell)}[c] = \sigma\left(\sum_{u \in \mathcal{N}(v)} \alpha_{c,vu}^{(\ell)} W^{(\ell)} h_u^{(\ell-1)}\right) [9]$$

dove σ è una funzione di attivazione, $W^{(\ell)}$ è una matrice di pesi appresa e $\alpha_{c,vu}^{(\ell)}$ rappresenta il coefficiente di attenzione appreso che quantifica il peso del messaggio proveniente dal nodo u verso il nodo v.
I vettori di **query** e **key**, necessari per calcolare le attenzioni, sono determinati come:

$$q_{c,v}^{(\ell)} = W_{c,q}^{(\ell)} h_v^{(\ell)} + b_{c,q}^{(\ell)}, \quad k_{c,u}^{(\ell)} = W_{c,k}^{(\ell)} h_u^{(\ell)} + b_{c,k}^{(\ell)} [9]$$

dove $W_{c,q}^{(\ell)}$ e $W_{c,k}^{(\ell)}$ sono matrici di pesi apprese, mentre $b_{c,q}^{(\ell)}$ e $b_{c,k}^{(\ell)}$ rappresentano i rispettivi bias.
Il coefficiente di attenzione $\alpha_{c,vu}^{(\ell)}$ è calcolato utilizzando un meccanismo di normalizzazione softmax e tiene conto anche delle informazioni sugli spigoli tramite un embedding $e_{c,vu}$:

$$\alpha_{c,vu}^{(\ell)} = \frac{\langle q_{c,v}^{(\ell)}, k_{c,u}^{(\ell)} + e_{c,vu} \rangle}{\sum_{w \in \mathcal{N}(v)} \langle q_{c,v}^{(\ell)}, k_{c,w}^{(\ell)} + e_{c,vw} \rangle} [9]$$

Questo schema permette al modello di adattare il peso dei messaggi in base sia alle caratteristiche dei nodi che alle proprietà delle connessioni tra di essi, migliorando la capacità di catturare relazioni strutturali complesse.
L'inclusione di un meccanismo di **attenzione multi-testa** consente di modellare simultaneamente diverse prospettive relazionali, stabilizzando il processo di apprendimento e migliorando la generalizzazione. Tutti i parametri, inclusi $W_{c,q}^{(\ell)}, W_{c,k}^{(\ell)}, b_{c,q}^{(\ell)}, b_{c,k}^{(\ell)}$, sono appresi durante l'addestramento. Questo approccio rende il **TransformerConv** particolarmente efficace per la classificazione semi-supervisionata sui grafi, permettendo una rappresentazione flessibile e informativa dei nodi e delle loro connessioni.

2.2 Implementazione

2.2.1 Creazione del grafo

Per la costruzione del grafo, si è implementato un grafo eterogeneo a partire dai dati relativi alle transazioni (*tx*) e ai portafogli (*wallet*), con l'obiettivo di modellare le relazioni tra questi elementi e facilitare l'apprendimento automatico su una struttura di rete. La rappresentazione eterogenea consente di distinguere tra diversi tipi di nodi e archi, migliorando la capacità del modello di catturare informazioni strutturali complesse.

La costruzione del grafo è avvenuta in più fasi: in primo luogo sono state estratte le informazioni sulle transazioni dai file *txs_edgelist.csv*, *txs_features.csv* e *txs_classes.csv*, mentre i dati relativi ai portafogli sono stati acquisiti dai file *AddrAddr_edgelist.csv* e *wallets_features_classes_combined.csv*. Per garantire la coerenza e l'affidabilità dei dati, sono state applicate diverse operazioni di pulizia:

- **Rimozione delle istanze con classe *unknown*:** poiché l'obiettivo del progetto è la classificazione binaria, le transazioni e i portafogli appartenenti a questa classe sono stati esclusi per evitare ambiguità nell'addestramento del modello.
- **Eliminazione di colonne non informative:** campi come *txId* e *Time step* che non contribuiscono all'apprendimento e per questo motivo sono stati rimossi da entrambi i dataset.
- **Filtraggio degli archi:** per garantire la consistenza del grafo, sono stati mantenuti solo gli archi in cui entrambi i nodi fossero presenti nel dataset elaborato.

Oltre alle connessioni interne ai nodi delle transazioni e dei portafogli, il grafo include archi intermodali che modellano le interazioni tra questi due livelli. Queste connessioni sono state estratte dai file *AddrTx_edgelist.csv* e *TxAddr_edgelist.csv*, e successivamente filtrate per includere solo nodi noti nel dataset finale. Tale approccio garantisce un'integrazione coerente tra le transazioni e i portafogli, migliorando la capacità del modello di apprendere le relazioni sottostanti.

Una volta completata la preparazione dei dati, il grafo è stato convertito nel formato PyTorch Geometric (PyG), definendo i seguenti tipi di nodi e relazioni:

- **Nodo *tx*:** rappresenta una transazione, caratterizzata da feature numeriche derivate dal dataset originale.
- **Nodo *wallet*:** rappresenta un portafoglio, con feature descrittive del comportamento finanziario.
- **Arco *is_related_to* (*tx* → *tx*):** connessioni dirette tra transazioni.
- **Arco *interacts_with* (*wallet* → *wallet*):** relazioni dirette tra portafogli.
- **Arco *performs* (*wallet* → *tx*):** connessione tra un portafoglio e una transazione che ha eseguito.
- **Arco *flows_into* (*tx* → *wallet*):** rappresenta una transazione i cui fondi confluiscono in un portafoglio.

Infine, il dataset è stato suddiviso in training (75%), validation (10%) e test (15%) utilizzando la funzione RandomNodeSplit, assicurando una distribuzione bilanciata per garantire la generalizzazione del modello.

2.2.2 Preprocessing

La fase di preprocessing ha l'obiettivo di minimizzare l'errore riducibile della funzione predittiva, migliorando così l'efficacia del modello.

Partendo dall'analisi delle feature, abbiamo osservato una disomogeneità tra le distribuzioni delle colonne, che indica la necessità di applicare tecniche di scaling o normalizzazione per uniformare i dati. Inoltre, considerando l'elevato numero di feature presenti nel dataset, abbiamo valutato l'impiego di tecniche di dimensionality reduction per ridurre la complessità del dataset e migliorare l'efficienza computazionale.

Per affrontare il problema della disomogeneità tra le distribuzioni delle feature, è stato necessario applicare tecniche di scaling e normalizzazione, selezionando la metodologia più adatta in base alla natura dei dati. Dall'analisi statistica di tx è emerso che le colonne local e aggregate risultavano già standardizzate, paragrafo 1.2.1. Per garantire coerenza e uniformità nel processo di preprocessing, si è quindi scelto di applicare la standardizzazione anche alle restanti feature del dataset. Nello specifico, è stata adottata **Standard Scaling**, implementata utilizzando la libreria scikit-learn, che trasforma le feature affinché abbiano media nulla e varianza unitaria, facilitando così la stabilità numerica del modello e migliorandone l'ottimizzazione.
Successivamente, si è valutata la possibilità di applicare una normalizzazione, come alternativa alla standardizzazione. Tuttavia, poiché la normalizzazione viene applicata sulle istanze e non sulle singole feature, in presenza di variabili con scale molto diverse, come nel nostro caso, avrebbe potuto introdurre distorsioni nei dati e compromettere la

qualità della trasformazione. Per risolvere questo problema, si è applicata una standardizzazione seguita dalla normalizzazione **L2**, che scala ciascuna riga del dataset affinché la sua norma euclidea sia pari a 1.

Per affrontare il problema dell'elevato numero di feature presenti nel dataset, è stato valutato di applicare la **Princial Component Analysis (PCA)** utilizzando la libreria scikit-learn. L'applicazione della PCA è stata sperimentata su entrambi i grafi, valutandone l'efficacia sia su dati standardizzati e/o normalizzati sia in assenza di pre-processing,.

2.2.3 Modelli

Per la classificazione dei nodi all'interno del grafo eterogeneo, è stata sviluppata un'architettura basata su una rete neurale convoluzionale eterogenea (Heterogeneous Graph Neural Network o HeteroGNN) utilizzando la libreria PyTorch Geometric (PyG) [3], implementando diverse tipologie di convoluzioni sui grafi, tra cui Graph Attention Networks (GAT), GraphSAGE (SAGE) e Transformer-based Convolutions (Transformer).
La rete è costituita da una serie di layer convoluzionali eterogenei che consentono di applicare operazioni diverse a seconda del tipo di relazione tra i nodi.

L'architettura del modello prevede un numero configurabile di strati convoluzionali(*num_layers*), ciascuno con un numero specificato di canali nascosti (*hidden_channels*). Durante il passaggio in avanti (forward pass), i nodi del grafo vengono trasformati attraverso le convoluzioni, seguite da una funzione di attivazione ReLU e un'operazione di dropout per prevenire overfitting. Per garantire una maggiore flessibilità, il modello supporta diverse strategie di aggregazione (*aggr*), che permettono di combinare le informazioni dei vicini in maniera differente.
Infine, il modello include due strati lineari separati (*lin_tx* e *lin_wallet*) per generare le previsioni finali, rispettivamente per i nodi di tipo tx e wallet. Il risultato è una rete in grado di apprendere rappresentazioni significative dai dati transazionali, migliorando l'accuratezza della classificazione e la capacità di identificare pattern nascosti nel comportamento finanziario.

Uno degli aspetti critici nell'addestramento del modello è la gestione dello squilibrio delle classi all'interno del dataset, che potrebbe portare a una riduzione della capacità del modello di classificare correttamente le classi meno rappresentate. Per mitigare questo problema, non sono state applicate tecniche di bilanciamento del dataset, come l'undersampling o l'oversampling. Al contrario, è stato adottato un approccio pesato nella funzione di perdita, calcolando i pesi delle classi in modo inversamente proporzionale alla loro frequenza nel dataset . Questi pesi vengono poi utilizzati all'interno della funzione di perdita *CrossEntropyLoss*, permettendo al modello di attribuire maggiore importanza agli esempi appartenenti alle classi meno rappresentate. Questo approccio consente di preservare la distribuzione originale dei dati, evitando distorsioni nella rete, e garantendo al contempo un migliore bilanciamento delle predizioni durante l'addestramento.

Capitolo 3

Risultati

3.1 Iperparametri testati

Per ottimizzare le prestazioni del modello, sono stati esplorati diversi set di iperparametri. Sono state sperimentate tre diverse architetture di convoluzione sui grafi: SAGEConv, GATConv, TransformerConv.

Iperparametro	Valori testati
Hidden Channels	32, 64, 128
Num Layers	2
Num Epoch	300
Patience	50
Learning Rate (LR)	0.001
Weight Decay	0, 1e-5, 5e-5
Dropout	0
Architettura	SAGEConv, GATConv, TransformerConv
Numero di Teste	2, 3, 4
p	5, 10
Factor	0.5, 0.2
Eta Min	1e-5
T_max	10, 15
Aggregazione	mean, sum
Learning Rate Scheduler	CosineAnnealingLR, ReduceLROnPlateau
Ottimizzatore	Adam

Tab. 3.1: Iperparametri testati durante la fase di addestramento.

La configurazione testata prevede un'architettura con due livelli (num_layers = 2) e un numero di hidden channels variabile tra 32, 64 e 128. Il modello è stato addestrato per un massimo di 300 epoche con una strategia di *early stopping* basata su una pazienza di 50 epoche.
L'ottimizzazione è stata effettuata utilizzando l'algoritmo Adam, mentre il learning rate è stato impostato a 0.001 con due possibili scheduler: CosineAnnealingLR e ReduceLROnPlateau. Inoltre, è stato valutato l'effetto del weight decay, testando i valori {0, 1e-5, 5e-5}. Nel caso di TransformerConv, il numero di teste di attenzione è stato testato tra 2, 3 e 4. È stata adottata una la Binary Cross Entropy (BCE) per il calcolo della loss, in quanto problema di classificazione binaria.
Per quanto riguarda il pre-processing, sono stati testati diversi approcci. In una prima fase, i risultati sono stati calcolati senza alcuna trasformazione sui dati. Successivamente, è stata valutata l'efficacia della standardizzazione e della normalizzazione. Infine, è

stato analizzato l'impatto della riduzione della dimensionalità applicando la PCA con una soglia fissata a 0.99, sia sui dati grezzi che su quelli preprocessati.
L'analisi delle prestazioni è stata condotta confrontando i risultati ottenuti nelle diverse configurazioni, al fine di valutare l'efficacia delle tecniche di pre-processing nel migliorare il bilanciamento tra accuratezza e complessità computazionale.

3.2 Scelta delle metriche di valutazione

Come già evidenziato nell'analisi delle feature (paragrafo 1.14), le classi del dataset risultano fortemente sbilanciate, con una netta predominanza della classe *legal* rispetto alla classe *illegal*. In situazioni di questo tipo, l'utilizzo dell'accuracy come metrica di valutazione può risultare inappropriato e fuorviante, poiché l'accuratezza tende a privilegiare la classe maggioritaria, trascurando le prestazioni di quella minoritaria. Questo comporta il rischio di ottenere valori elevati di accuracy anche quando il modello è incapace di riconoscere correttamente le istanze della classe meno rappresentata, portando a una valutazione eccessivamente ottimistica delle capacità predittive del modello.
Per affrontare questo problema, si è preferito utilizzare la *balanced accuracy* o, in alternativa, l'*F1-score* come metriche principali per la valutazione delle prestazioni del modello. La balanced accuracy rappresenta la media tra la sensibilità (recall) calcolata separatamente per ciascuna classe, garantendo così una valutazione più equa anche in presenza di squilibri significativi tra le classi. In altre parole, questa metrica considera in modo bilanciato l'accuratezza ottenuta su ciascuna classe, evitando che la predominanza della classe *legal* domini il risultato complessivo. D'altro canto, l'F1-score è definito come la media armonica tra precision e recall. Questa metrica risulta particolarmente utile quando è cruciale bilanciare il numero di falsi positivi e falsi negativi, poiché combina la capacità di identificare correttamente gli esempi positivi (recall) con la precisione nell'evitare di includere esempi negativi (precision).
L'adozione della balanced accuracy e dell'F1-score consente pertanto di ottenere una valutazione più affidabile e rappresentativa delle prestazioni del modello, garantendo un'analisi più bilanciata e accurata.

3.3 SAGE

3.3.1 Classificazione delle transazioni

Dall'analisi dei risultati emerge che il preprocessing dei dati ha un impatto significativo sulle prestazioni dell'architettura Sage, valutate sulla base dell'accuratezza bilanciata per classe. In particolare, l'uso della standardizzazione e normalizzazione combinata con la riduzione della dimensionalità tramite PCA si dimostra altamente efficace, migliorando le performance rispetto ai dati grezzi. Le configurazioni standard-pca e standard_l2-pca raggiungono valori di accuratezza pesata media superiori a 0.88, con un massimo di 0.900 in combinazione con 128 hidden channels e aggregazione sum. Questo suggerisce che la riduzione della dimensionalità non solo aiuta a eliminare il rumore, ma fornisce anche una rappresentazione più compatta e informativa dei dati, facilitando l'apprendimento del modello. Al contrario, l'assenza di preprocessing (no-no) porta a prestazioni significativamente inferiori, con accuratezze intorno a 0.56-0.58. Anche la standardizzazione

senza PCA (standard-no e standard_l2-no) offre un miglioramento rispetto alla mancanza di preprocessing, ma non raggiunge le performance delle configurazioni che includono PCA.

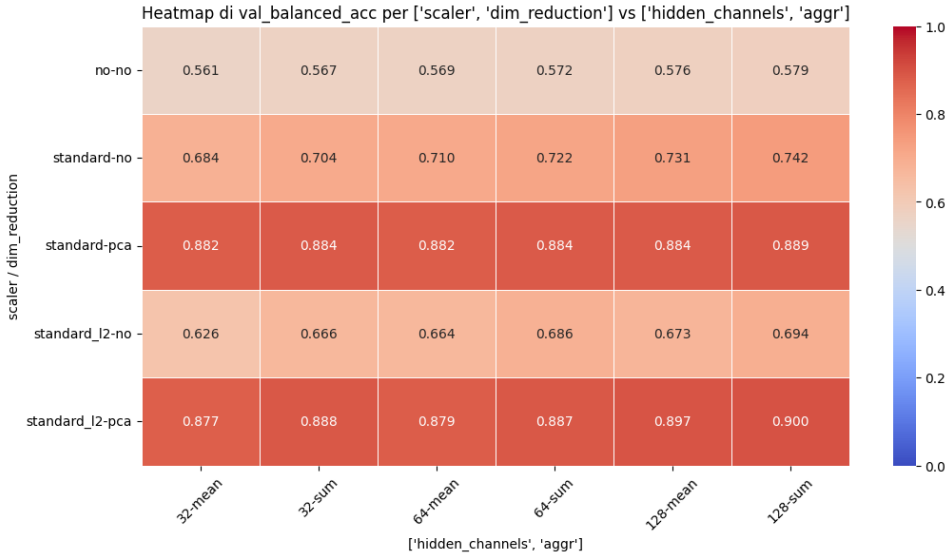

Fig. 3.1: Heatmap dell'accuratezza bilanciata media per la classificazione di tx

Nella Figura 3.1 è riportata una heatmap che mostra l'accuratezza pesata media in funzione del preprocessing applicato (scaler e dimensionality reduction) e della combinazione tra hidden channels e tecnica di aggregazione. Si osserva chiaramente che le combinazioni che includono la PCA ottengono le migliori prestazioni, con una tendenza generale di miglioramento all'aumentare del numero di hidden channels.
Analizzando invece l'impatto del learning rate scheduler tramite il boxplot della Figura 3.2, si osserva che la strategia ReduceLROnPlateau tenda a fornire risultati migliori rispetto a CosineAnnealingLR. Sebbene gli estremi di entrambe le distribuzioni siano vicini, la mediana delle prestazioni con ReduceLROnPlateau risulta più elevata, suggerendo una maggiore stabilità nell'ottimizzazione e un miglior adattamento della rete al problema. Questo risultato potrebbe essere attribuito alla capacità di ReduceLROnPlateau di ridurre dinamicamente il learning rate in risposta al mancato miglioramento della loss, permettendo al modello di convergere in modo più efficace.

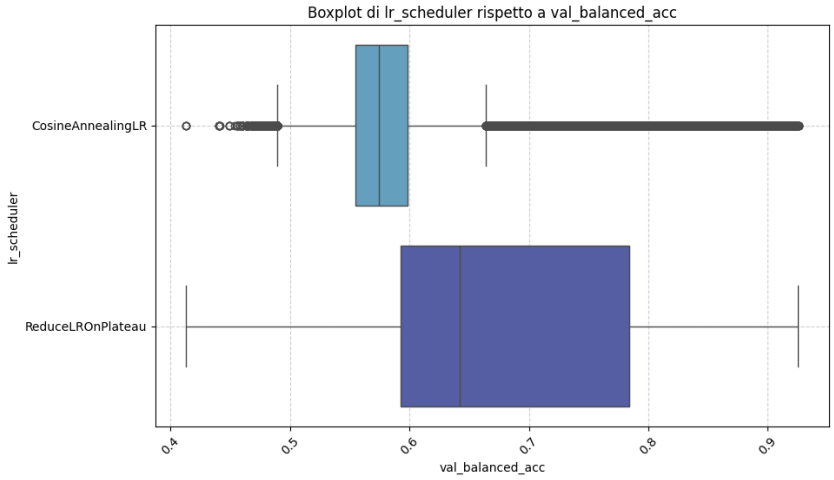

Fig. 3.2: Boxplot dell'accuratezza bilanciata sul *lr_scheduler* di tx

Migliore Configurazione di Iperparametri Per la classificazione delle transazioni, è stata utilizzata un'architettura *HeteroGNN* con convoluzione *SAGE* e aggregazione *mean*. Il modello è stato addestrato per *300 epoche*, con un criterio di early stopping basato su una pazienza di *50 epoche*. La rete utilizza *128 hidden channels* e un learning rate iniziale di *0.001*, regolato dinamicamente tramite lo scheduler *ReduceLROnPlateau* con fattore di riduzione pari a *0.5*. Per evitare overfitting, è stato applicato un *weight decay* di $1e^{-5}$, mentre non è stato introdotto dropout. Inoltre, è stata adottata una strategia di preprocessing basata sulla *standardizzazione con norma L2* e sulla *riduzione dimensionale tramite PCA*, con una soglia di varianza pari al *99%*.

Set	Loss	Accuracy	F1-score	Balanced Acc.
Train	0.0179	0.9947	0.9971	0.9971
Validation	0.3174	0.9706	0.9837	0.9238
Test	0.3480	0.9664	0.9813	0.9129

Tab. 3.2: Prestazioni del modello SAGE sui diversi set per la classificazione delle transazioni.

L'analisi dei risultati mostra che il modello raggiunge prestazioni eccellenti sul training set, con un'accuratezza del 99.47% e un F1-score di 0.9971, indicando che la rete è in grado di apprendere molto bene i pattern nei dati di addestramento. Tuttavia, osservando la differenza tra i risultati su training e validation/test set, emerge una leggera varianza nelle performance. In particolare, sul validation set l'accuratezza scende al 97.06%, e il balanced accuracy diminuisce a 92.38%, mentre sul test set si registra un'ulteriore lieve flessione (accuracy 96.64%, balanced accuracy 91.29%). Questo suggerisce che il modello potrebbe essere leggermente overfit rispetto ai dati di training, pur mantenendo una forte capacità di generalizzazione.

Classe	Precision	Recall	F1-score
Classe 0 (Val)	0.84	0.87	0.85
Classe 1 (Val)	0.99	0.98	0.98
Classe 0 (Test)	0.81	0.85	0.83
Classe 1 (Test)	0.98	0.98	0.98

Tab. 3.3: Metriche di precision, recall e F1-score per ogni classe.

Dall'analisi delle metriche per classe 3.3, emerge un'ottima capacità del modello nel riconoscere la classe maggioritaria (1), con valori di precisione e recall prossimi a 1. Tuttavia, la classe minoritaria (0) registra prestazioni inferiori, un risultato atteso data la forte asimmetria nella distribuzione delle istanze nel dataset di addestramento.

3.3.2 Wallet

L'analisi delle prestazioni per la classificazione del wallet conferma il forte impatto del preprocessing sui risultati del modello Sage. Come evidenziato nella heatmap della Figura 3.3, le configurazioni che includono la riduzione dimensionale con PCA ottengono le migliori accuratezze bilanciate, con valori che raggiungono 0.850 nella combinazione standard_l2-pca con 128 hidden channels e aggregazione sum. Questo conferma il ruolo chiave della PCA nel miglioramento delle performance. D'altra parte, l'assenza di preprocessing (no-no) porta a prestazioni inferiori, con accuratezze comprese tra 0.68 e 0.69, mentre la sola standardizzazione senza PCA (standard-no e standard_l2-no) mostra un

leggero miglioramento, sebbene rimanga distante dalle configurazioni che includono PCA. Anche in questo caso, la tendenza suggerisce che una rappresentazione più compatta e informativa dei dati consenta al modello di apprendere in modo più efficace.

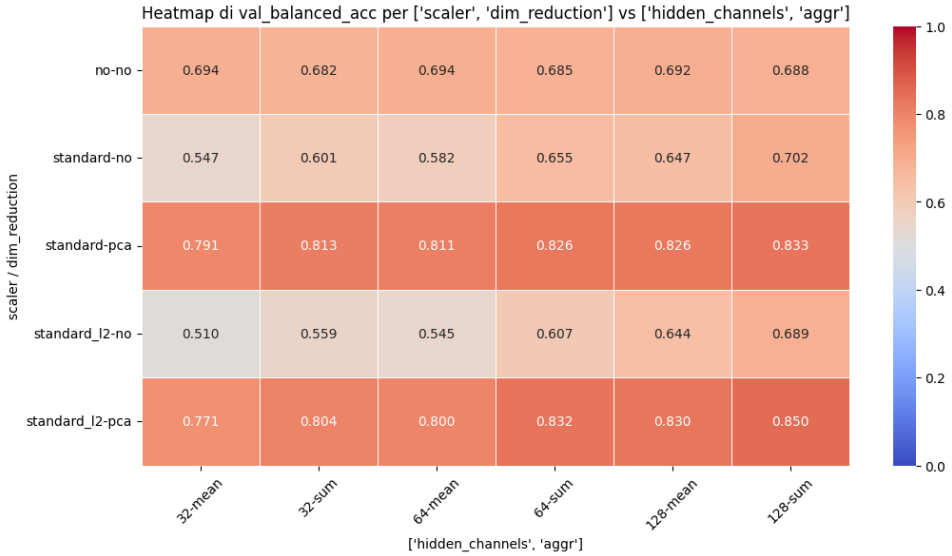

Fig. 3.3: Heatmap dell'accuratezza bilanciata media per la classificazione di wallet

Migliore configurazione di Iperparamentri Il modello addestrato per la classificazione dei wallet è stato configurato con un'architettura *HeteroGNN* basata su convoluzioni *SAGE*, utilizzando *128 hidden channels* e *2 livelli*. L'ottimizzazione è stata eseguita con *Adam*, con un tasso di apprendimento iniziale di *0.001* e un peso di regolarizzazione *L2 pari a 1e-5*. Il modello ha seguito un apprendimento di *300 epoche*, con un criterio di *early stopping a 50 epoche di pazienza*. È stato impiegato il scheduler *CosineAnnealingLR* con *T_max=15*, mentre la strategia di aggregazione adottata è stata *sum*. Inoltre, è stata applicata una riduzione dimensionale tramite *PCA*, mantenendo il *99% della varianza* e normalizzando i dati con lo *standard scaler L2*. Al termine dell'addestramento, il modello ha ottenuto ottimi risultati su tutti i set di dati. In particolare, sul validation set ha raggiunto una F1-score di **0.9451** con un'accuratezza del **90.3%** e un balanced accuracy di **88.39%**. Sul test set, le performance rimangono stabili, con F1-score di **0.9440**, accuratezza del **90.1%** e balanced accuracy di **87.92%**.

Set	Loss	Accuracy	F1-score	Balanced Acc.
Train	0.221	0.9111	0.9498	0.9121
Val	0.277	0.9030	0.9451	0.8839
Test	0.280	0.9011	0.9440	0.8792

Tab. 3.4: Performance del modello SAGE per la classificazione di wallet.

L'analisi per classe evidenzia una netta superiorità nella classificazione della classe maggioritaria (1), con una precisione e recall molto elevati. La classe minoritaria (0) risulta più difficile da classificare correttamente, con una recall attorno a **85-86%** e un **F1-score intorno a 0.58**. Questo effetto è attribuibile allo sbilanciamento del dataset, dove la classe minoritaria è significativamente meno rappresentata.

Classe	Precision	Recall	F1-score
Classe 0 (Val)	0.43	0.85	0.58
Classe 1 (Val)	0.99	0.91	0.94
Classe 0 (Test)	0.44	0.86	0.58
Classe 1 (Test)	0.99	0.91	0.95

Tab. 3.5: Metriche di precision, recall e F1-score per ogni classe.

3.4 GAT

3.4.1 Classificazione delle transazioni

Dall'analisi dei risultati emerge come l'architettura GAT, nel contesto del dataset considerato, non riesca a ottenere prestazioni competitive per la classificazione delle transazioni illecite. In particolare, osserviamo che la distribuzione dei valori di accuratezza bilanciata è molto limitata, con i risultati concentrati attorno al 50%. Questo comportamento suggerisce che il modello fatichi a distinguere efficacemente tra le classi, risultando fortemente influenzato dalla classe dominante, il che ne compromette la capacità di generalizzazione. Tuttavia, si riscontrano lievi miglioramenti quando vengono applicate tecniche di preprocessing, in particolare con la riduzione della dimensionalità tramite PCA. Come evidenziato dal boxplot in Figura 3.6, l'adozione della PCA contribuisce a una leggera crescita della mediana della accuratezza bilanciata rispetto alla configurazione senza riduzione della dimensionalità. Nonostante ciò, il miglioramento non è sufficiente per garantire prestazioni soddisfacenti.
Un altro fattore che sembra incidere, seppur in maniera marginale, è la scelta del learning rate scheduler. Come mostrato in Figura 3.7, l'utilizzo di *ReduceLROnPlateau* risulta in una distribuzione dell'accuratezza bilanciata leggermente migliore rispetto al *CosineAnnealingLR*. Questo suggerisce che l'adattamento dinamico del learning rate possa fornire un contributo positivo, anche se limitato, alla stabilità del training.
Complessivamente, sebbene alcuni accorgimenti migliorino leggermente le performance del modello, i risultati indicano che il GAT non è in grado di apprendere efficacemente le rappresentazioni necessarie per ottenere buone prestazioni su questo task.

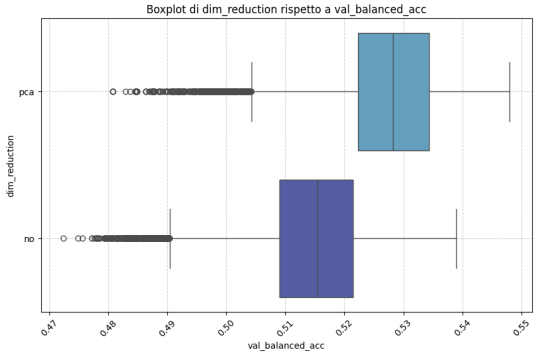

Fig. 3.4: Boxplot dell'accuratezza bilanciata sul *dim_reduction* di tx

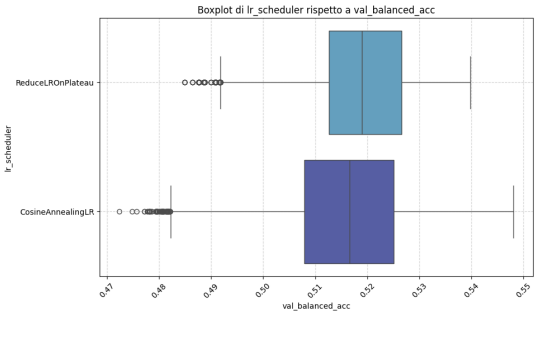

Fig. 3.5: Boxplot dell'accuratezza bilanciata sul *lr_scheduler* di tx

Migliore Configurazione di Iperparametri Analizzando i risultati 3.6 si è ottenuto che la la configurazione ottimale del modello per la classificazione delle transazioni

prevede *128 hidden channels*, *2 layer*, *77 epoche* e un meccanismo di early stopping con *patience pari a 50*. Il modello utilizza un *learning rate* di 0.001, con *weight decay* pari a ($1e^{-5}$, senza dropout, utilizzano una convoluzione di tipo *GAT*. L'ottimizzazione avviene con l'algoritmo *Adam*, mentre la regolazione del learning rate segue la strategia *CosineAnnealingLR*, con un $\eta_{\min}$ di $1e^{-5}$ e $T_{\max} = 10$. E l'aggregazione è stata effettuata tramite operazione di *sum*.
I risultati ottenuti indicano che il modello non è sufficientemente espressivo per catturare la complessità del dataset. L'accuratezza sul training set è **69.58%**, mentre sul validation set scende al **67.20%**, con un balanced accuracy di soli **55.08%**, segnalando difficoltà nel distinguere correttamente le classi. I risultati sul test set sono molto simili a quelli di validazione (**67.25%** di accuratezza), indicando che non vi è una forte varianza nel modello, ma piuttosto un bias elevato, dovuto probabilmente a una capacità di rappresentazione insufficiente o a una scelta subottimale degli iperparametri.

Fase	Loss	Accuracy	F1-score	Balanced Accuracy
Train	0.6082	0.6958	0.8081	0.6322
Validation	0.7133	0.6720	0.7940	0.5508
Test	0.7017	0.6725	0.7936	0.5691

Tab. 3.6: Risultati del modello GAT per la classificazione delle transazioni.

In particolare, dalla valutazione per classe emerge che la classe minoritaria (0) è classificata con un F1-score molto basso (0.19 sul validation set, 0.21 sul test set) e una precisione inferiore al 15%, evidenziando una forte difficoltà nel bilanciare le due classi (Tab. 3.7). Questo risultato suggerisce che il modello potrebbe beneficiare di una maggiore complessità architetturale.

Classe	Precision	Recall	F1-score
Classe 0 (Val)	0.13	0.40	0.19
Classe 1 (Val)	0.91	0.70	0.79
Classe 0 (Test)	0.14	0.44	0.21
Classe 1 (Test)	0.92	0.70	0.79

Tab. 3.7: Metriche di precision, recall e F1-score per ogni classe.

3.4.2 Wallet

L'analisi per la classificazione dei wallet, i valori di accuratezza bilanciata sulle diverse configurazioni mostrano tendenze simili a quelle osservate per le transazioni.
Dai risultati emerge che l'uso dello scaler standard e lo scheduler cosenico migliorano le performance del modello (Fig. 1.13). La riduzione dimensionale tramite PCA non porta significativi miglioramenti (Fig. 1.12), con performance simili o leggermente inferiori rispetto alle configurazioni senza PCA. In generale, le diverse combinazioni di hidden_channels e aggr non influenzano in modo significativo le prestazioni.

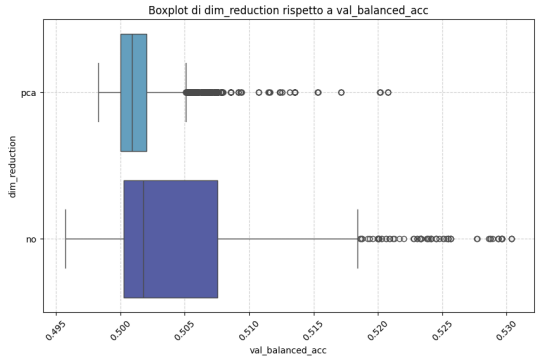

Fig. 3.6: Boxplot dell'accuratezza bilanciata sul *dim_reduction* di wallet

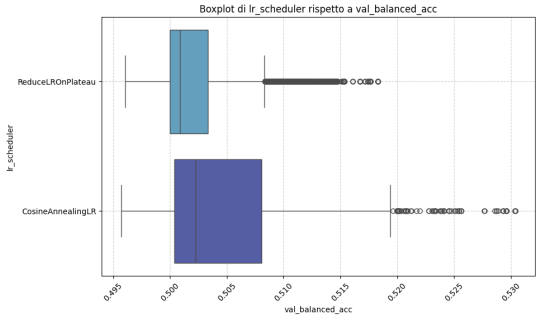

Fig. 3.7: Boxplot dell'accuratezza bilanciata sul *lr_scheduler* di wallet

Migliore Configurazione di Iperparametri La migliore configurazione di Graph Attention Networks (GAT) per la classificazione dei wallet è stata ottenuta con 64 hidden channel e 2 livelli, utilizzando l'ottimizzatore Adam con un tasso di apprendimento di 0.001. L'addestramento è stato interrotto dopo 90 epoche, con una strategia di CosineAnnealingLR (T_max=15) per la regolazione del tasso di apprendimento. A differenza di altre configurazioni, qui non è stata applicata una riduzione dimensionale e la normalizzazione è stata effettuata con lo standard scaler. L'aggregazione scelta per il modello è stata sum.

Fase	Loss	Accuracy	F1-score	Balanced Accuracy
Train	0.7524	0.6470	0.7778	0.5226
Val	0.7479	0.6529	0.7822	0.5258
Test	0.7527	0.6447	0.7761	0.5182

Tab. 3.8: Metriche di Performance del modello GAT per la classificazione dei wallet.

Le prestazioni del modello evidenziano una forte difficoltà nel distinguere le classi, con un balanced accuracy che si attesta intorno al **52%** su validation e test set. Sebbene l'F1-score globale sia moderatamente buono (**0.78** circa su tutti i set), il modello soffre di un forte sbilanciamento nelle predizioni, classificando in modo efficace solo la classe maggioritaria (1).

Classe	Precision	Recall	F1-score
Classe 0 (Val)	0.09	0.37	0.1
Classe 1 (Val)	0.93	0.68	0.78
Classe 0 (Test)	0.09	0.37	0.14
Classe 1 (Test)	0.93	0.67	0.78

Tab. 3.9: Metriche di precision, recall e F1-score per ogni classe.

Dai risultati per classe (Tab. 3.9), emerge chiaramente che la classe minoritaria (0) è identificata con una precisione bassissima (circa 9%) e un recall del 37%, portando a un F1-score inferiore a 0.15. Questo indica che il modello ha un forte bias verso la classe predominante, ignorando quasi completamente la classe meno rappresentata.
Inoltre, il fatto che le metriche siano molto simili tra train, validation e test set suggerisce che il modello soffra di alto bias.

3.5 Transformer

3.5.1 Classificazione delle transazioni

L'analisi delle prestazionisui livelli di convoluzione che utilizzando Transformer rivela un impatto significativo delle scelte relative agli iperparametri sulla balanced accuracy, queste sono riportate nella heatmap in Figura 3.8 che dimostra come il learning rate scheduling e la strategia di aggregazione influenzino in maniera sostanziale l'efficacia del modello. Configurazioni che impiegano la somma (sum) come metodo di aggregazione e lo scheduling del learning rate tramite *ReduceLROnPlateau* raggiungono le performance migliori, con valori di balanced accuracy superiori a 0,90.
Anche il numero di hidden channels gioca un ruolo significativo, infine un ulteriore elemento di influenza è il numero di teste nei livelli di self-attention: valori più bassi (come 2 o 3) producono risultati migliori, mentre con 4 teste si ottengono performance inferiori. Tuttavia, è importante sottolineare che non è stato possibile testare configurazioni con 4 teste e 64 hidden channels a causa delle limitazioni di risorse di Google Colab.

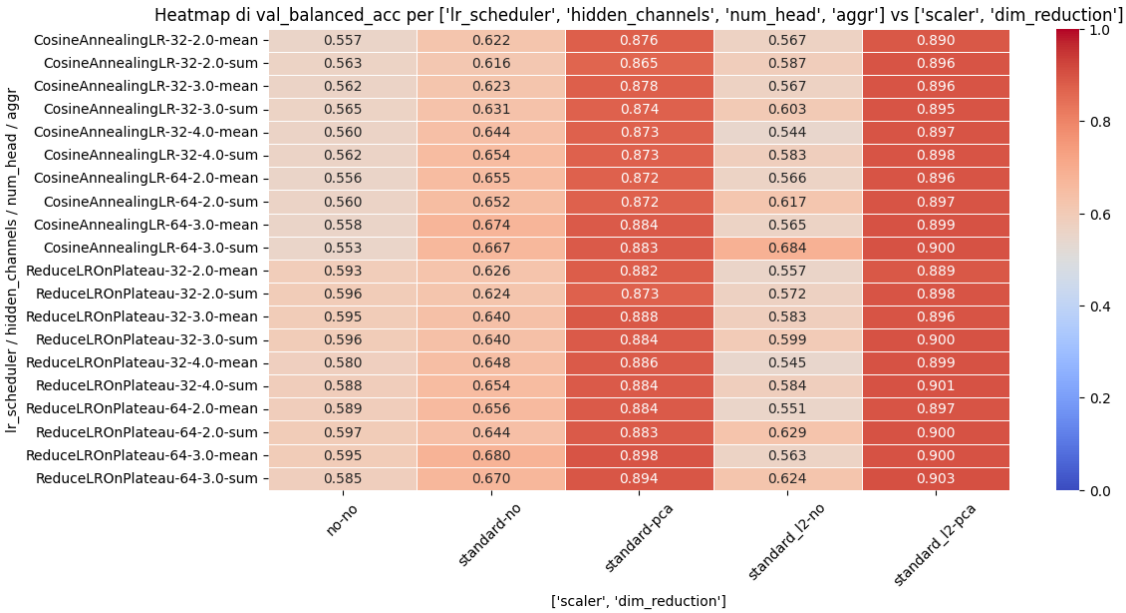

Fig. 3.8: Heatmap dell'accuratezza bilanciata media per la classificazione di tx

Dall'analisi emerge chiaramente l'importanza del preprocessing dei dati, le tecniche di riduzione dimensionale associate alla standardizzazione hanno permesso di ottenere accuratezze pesate superiori a 0,86, evidenziando l'efficacia di un approccio che riduce il rumore e migliora la compattezza dei dati: in particolare, l'uso combinato di standardizzazione, normalizzazione e riduzione della dimensionalità tramite PCA ha portato a un miglioramento delle performance del 40% rispetto ai dati grezzi. Al contrario, l'assenza di preprocessing ha prodotto performance notevolmente inferiori (accuratezze tra 0,55 e 0,59).

I risultati relativi alla scelta degli iperparametri sono chiaramente visibili nei boxplot della Figura 3.9, dai quali si osserva che le diverse strategie di scheduling del learning non comportano cambiamenti significativi nelle performance, mostrando distribuzioni, estremi e mediane comparabili. Si nota invece che l'applicazione della riduzione dimensionale e delle tecniche di scaling ha un impatto positivo sulle prestazioni del modello.

L'assenza di scaling produce risultati più compatti, ma con performance inferiori, mentre l'uso esclusivo della standardizzazione genera prestazioni superiori. Infine, la strategia più efficace si conferma la combinazione di standardizzazione e normalizzazione, che, pur mantenendo estremi comparabili alla sola standardizzazion, presenta una mediana superiore rispetto a quest'ultima.

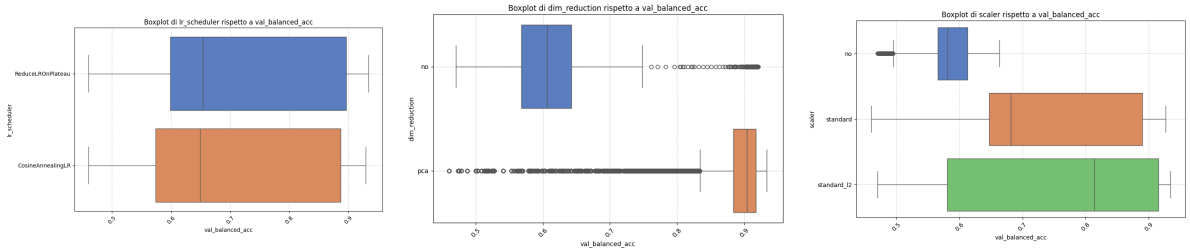

Fig. 3.9: Boxplot dell'accuratezza bilanciata rispetto ai *lr_scheduler*, *dim_reduction* e *scaler* di tx

Migliore Configurazione di Iperparametri Per la classificazione delle transazioni, è stata adottata un'architettura *HeteroGNN* con convoluzione *Transformer* e aggregazione *mean*. Il modello è stato addestrato per un totale di *300 epoche*, con una strategia di early stopping che ha previsto una pazienza di *50 epoche*. La rete impiega *32 hidden channels* e un learning rate iniziale di *0.001*, che viene adattato dinamicamente tramite lo scheduler *ReduceLROnPlateau* con un fattore di riduzione di *0.5*. Per contrastare il rischio di overfitting, è stato utilizzato un *weight decay* pari a $1e^{-5}$, mentre non è stato applicato dropout. Inoltre, il preprocessing include una *standardizzazione con norma L2* e una *riduzione dimensionale tramite PCA*, con un criterio di varianza fissato al *99%*.

Set	**Loss**	**Accuracy**	**F1-score**	**Balanced Acc.**
Train	0.0691	0.9786	0.9880	0.9834
Validation	0.1788	0.9603	0.9777	0.9328
Test	0.1826	0.9580	0.9765	0.9251

Tab. 3.10: Prestazioni del modello Transformer sui diversi set per la classificazione delle transazioni.

I risultati mostrano che il modello ottiene ottime prestazioni sul set di addestramento, con una balanced accuracy del 98.34% e un F1-score pari a 0.9880, indicando un'efficace capacità di apprendimento dei pattern nei dati di training. Tuttavia, confrontando le performance sui set di training, validation e test, si nota una leggera variazione nei risultati. In particolare, sul set di validation l'F1-score scende al 97,77% e l'accuratezza bilanciata diminuisce al 93,28%, mentre sul test set si osserva ancora una flessione, seppur più lieve che vede un F1-score al 97,65% e un'accuratezza bilanciata al 92,51%. Anche questi risultati come quelli del SAGE suggeriscono che il modello potrebbe essere leggermente overfit sui dati di training, pur mantenendo una buona capacità di generalizzazione.

Classe	**Precision**	**Recall**	**F1-score**
Classe 0 (Val)	0.75	0.90	0.82
Classe 1 (Val)	0.99	0.97	0.98
Classe 0 (Test)	0.74	0.88	0.80
Classe 1 (Test)	0.99	0.97	0.98

Tab. 3.11: Metriche di precision, recall e F1-score per ogni classe.

Dall'analisi delle metriche per ciascuna classe, tabella 3.11, si evidenzia una notevole capacità del modello nel riconoscere la classe prevalente (1), con valori di precisione e recall molto vicini a 1. Al contrario, le prestazioni per la classe meno rappresentata (0) risultano inferiori, un esito previsto considerando la marcata asimmetria nella distribuzione delle istanze nel dataset di addestramento.

3.5.2 Wallet

L'analisi per la classificazione dei wallet, effettuata sulle diverse configurazioni, mostra tendenze simili a quelle osservate per le transazioni. Inoltre, conferma quanto già rilevato nei precedenti modelli, ovvero che le performance relative a tx sono superiore a quelle registrate per i wallet.

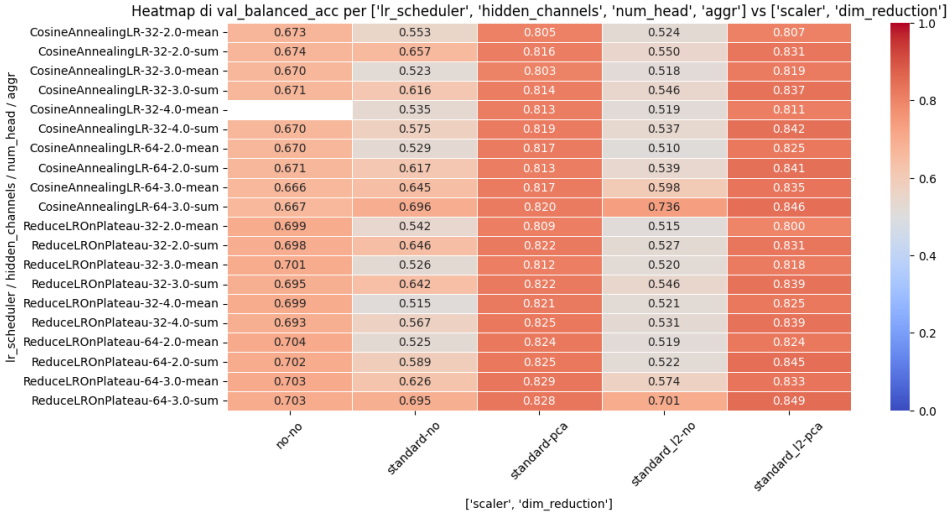

Fig. 3.10: Heatmap dell'accuratezza bilanciata media per la classificazione di wallet.

Analogamente a quanto osservato in tx sui livelli di convoluzione, l'uso dei Transformer risente in modo significativo della strategia di aggregazione e del numero di hidden channels. Anche in questo caso, emerge chiaramente che la riduzione della dimensionalità tramite PCA risulta particolarmente efficace, portando a un miglioramento delle performance di quasi il 30% rispetto all'uso dei dati grezzi. Questo risultato indica che la riduzione della dimensionalità non solo contribuisce alla riduzione del rumore nei dati, ma offre anche una rappresentazione più compatta e informativa, facilitando l'apprendimento del modello.
Mentre, al contrario di quanto osservato per tx, le strategie di standardizzazione hanno comportato un calo delle performance rispetto all'analisi dei dati grezzi, raggiungendo un'accuratezza bilanciata intorno al 0,55 nel caso della standardizzazione e valori leggermente inferiori per la standardizzazione combinata con la normalizzazione L2. Anche il numero di head nei livelli di self-attention incide sulle prestazioni, sebbene non in modo particolarmente significativo: valori più bassi, come 2 e 3, hanno fornito risultati migliori, mentre con 4 si osservano performance inferiori. È tuttavia bene notare che come già avvenuto per tx non è stato possibile testare modelli con 4 head e 64 hidden layer a causa delle risorse limitate disponibili su Colab.

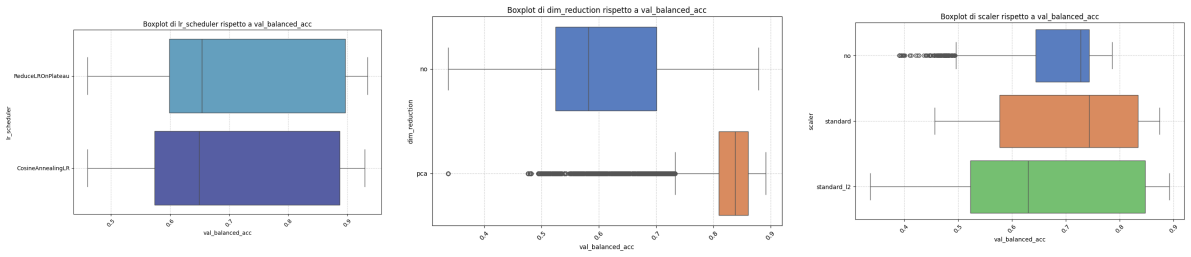

Fig. 3.11: Boxplot dell'accuratezza bilanciata rispetto ai *lr_scheduler*, *dim_reduction* e *scaler* di wallet

Le osservazioni riportate sono evidenti anche nei boxplot mostrati in Figura 3.11. Si può osservare che lo scheduler dinamico del learning rate, come già evidenziato per il caso TX (Figura 3.9), non produce una variazione significativa delle performance.
Miglioramenti significativi sono invece derivante dall'uso della PCA e tecniche di scaling: l'assenza dello scheduler determina una distribuzione più compatta dei valori, ma con prestazioni inferiori, mentre utilizzando esclusivamente la standardizzazione, i risultati si distribuiscono su un intervallo più ampio, con performance superiori; questo comportamento è dovuto alla valutazione di modelli che presentano standardizzazione con l'assenza della PCA, che produce come visto nell'heatmap **??** prestazioni inferiori e alla valutazioni di modelli che presentano standardizzazione con la PCA.
Quando la standardizzazione viene combinata con la normalizzazione, i risultati mostrano un intervallo ancora più esteso, con performance complessivamente migliori, sebbene la mediana risulti leggermente inferiore rispetto all'uso esclusivo della standardizzazione.

Migliore Configurazione di Iperparametri Il modello addestrato per la classificazione dei wallet è stato configurato con un'architettura *HeteroGNN* basata su convoluzioni *Transformer*, impiegando *64 canali nascosti*, *2 layer* e *2 teste*. Per l'ottimizzazione, è stato utilizzato l'algoritmo *Adam* con un tasso di apprendimento iniziale di *0.001* e un *peso di regolarizzazione L2 pari a 1e-5*. Il processo di addestramento ha seguito un ciclo di *300 epoche*, con un criterio di early stopping impostato a *50* epoche di pazienza per prevenire fenomeni di overfitting.
Per quanto riguarda la gestione del tasso di apprendimento, è stato utilizzato lo *scheduler ReduceLROnPlateau* con parametri *p=10* e *factor=0.2*, al fine di ridurre il learning rate in caso di stagnazione delle performance. La strategia di aggregazione scelta è stata *sum*, garantendo una combinazione efficace delle caratteristiche estratte. Inoltre, è stata effettuata una riduzione dimensionale tramite PCA, preservando il 99% della varianza e normalizzando i dati tramite lo *Standard Scaler ed L2*.

Set	Loss	Accuracy	F1-score	Balanced Acc.
Train	0.2419	0.9059	0.9467	0.9037
Val	0.2618	0.9031	0.9451	0.8919
Test	0.2597	0.9018	0.9443	0.8916

Tab. 3.12: Prestazioni del modello Transformer sui diversi set per la classificazione dei wallet.

Al termine dell'addestramento, il modello ha raggiunto ottimi risultati su tutti i set di dati. In particolare, sul validation set ha ottenuto un F1-score di 0.9451, un'accuratezza del 90.31% e una balanced accuracy di 89.199%. Tali valori, oltre a rispecchiare i risultati ottenuti durante la fase di training, si avvicinano alle performance registrate con l'architettura SAGE, risultato che potrebbe indicare una complessità intrinseca nella

struttura dei dati stessi o una sovrapposizione parziale tra le classi, rendendo difficoltosa una separazione completamente accurata.
Anche sul test set, le prestazioni del modello si mantengono stabili, con un F1-score di 0.9443, un'accuratezza del 90.18% e una balanced accuracy di 89.16%. La coerenza dei risultati tra il validation set e il test set conferma la buona generalizzazione del modello, dimostrando che l'approccio adottato riesce a mantenere elevate le performance anche su dati non visti durante l'addestramento.

Classe	**Precision**	**Recall**	**F1-score**
Classe 0 (Val)	0.44	0.88	0.59
Classe 1 (Val)	0.99	0.91	0.95
Classe 0 (Test)	0.44	0.88	0.59
Classe 1 (Test)	0.99	0.90	0.94

Tab. 3.13: Metriche di precision, recall e F1-score per ogni classe.

Nuovamente l'analisi per classe evidenzia una netta superiorità nella classificazione della classe maggioritaria (1), con una precisione e recall molto elevati, mentre la classe minoritaria (0) risulta più difficile da classificare correttamente, con una recall attorno a **88%** e un **F1-score intorno a 0.59**. Questo effetto è attribuibile allo sbilanciamento del dataset, dove la classe minoritaria è significativamente meno rappresentata.

3.6 Risultati complessivi

La valutazione comparativa dei modelli implementati, sintetizzata nella tabella 3.14, evidenzia come il modello *transformConv* abbia conseguito le prestazioni superiori in termini di balanced accuracy. Tuttavia, l'analisi rivela una discrepanza statisticamente significativa tra le performance ottenute durante la fase di training e quelle riscontrate nelle fasi di validazione e test. Questa divergenza, quantificabile in diversi punti percentuali, suggerisce la presenza di una varianza nel modello, indicativa di un potenziale sovra-adattamento ai dati di training. Tale fenomeno può essere attribuito a una rappresentazione subottimale della classe delle transazioni illecite all'interno del dataset, che risulta sottorappresentata.

		TX Metrics			**Wallet Metrics**		
		Acc.	**F1**	**Bal Acc.**	**Acc.**	**F1**	**Bal Acc.**
SAGE	Train	99.47%	0.9971	99.71%	91.11%	0.9498	91.21%
	Val	97.06%	0.9837	92.38%	90.30%	0.9451	88.39%
	Test	**96.64%**	**0.9813**	91.29%	90.11%	0.9440	87.92%
GAT	Train	69.58%	0.8081	63.22%	64.70%	0.7778	52.26%
	Val	67.20%	0.7940	55.08%	65.29%	0.7822	52.58%
	Test	67.25%	0.7936	56.91%	64.47%	0.7761	51.82%
Transf	Train	97.86%	0.9880	98.34%	90.59%	0.9467	90.37%
	Val	96.03%	0.9777	93.28%	90.31%	0.9451	89.19%
	Test	95.80%	0.9765	**92.51%**	**90.18%**	**0.9443**	**89.16%**

Tab. 3.14: Prestazioni riassuntive della classificazione delle transazioni e dei Wallet al variare dei modelli.

L'analisi dell'andamento delle performance in funzione del numero di epoche, estesa fino a 1000 iterazioni, ha fornito ulteriori spunti di riflessione. Nel contesto della classificazione delle transazioni (Fig. 3.12), si è osservato che la funzione di loss relativa al training continua a decrescere asintoticamente verso lo zero. Parallelamente, la loss di validazione mostra un andamento crescente dopo circa 340 epoche, indicando l'insorgenza di un fenomeno di overfitting. Tale comportamento è corroborato dall'analisi dell'errore, che segue un andamento analogo.

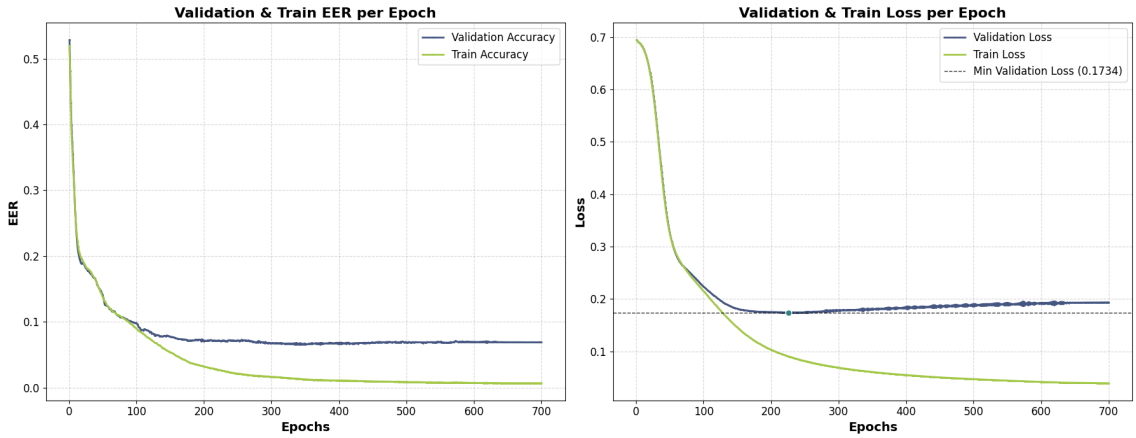

Fig. 3.12: Andamento dell'EER e della loss al variare dell'epoca sul miglior modello di classificazione di tx

Al contrario, il modello di classificazione dei wallet non manifesta segni di overfitting (Fig. 3.13). In questo caso, entrambe le curve di loss (training e validazione) continuano a decrescere, seppur con una pendenza inferiore rispetto alle epoche iniziali. L'epoca ottimale, identificata alla 828ª iterazione, suggerisce che il modello è in grado di apprendere ulteriormente senza compromettere la capacità di generalizzazione.

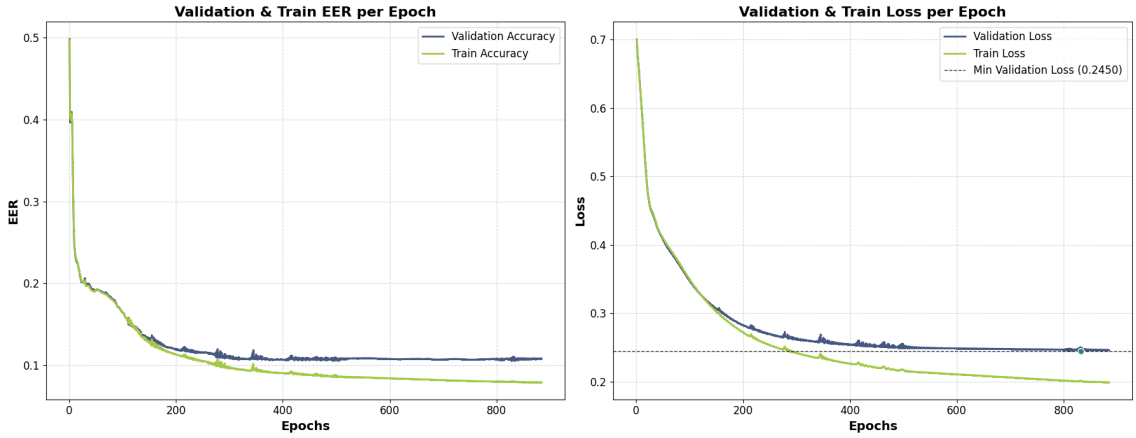

Fig. 3.13: Andamento dell'EER e della loss al variare dell'epoca sul miglior modello di classificazione di wallet

Un aspetto chiave emerso da questa analisi è l'importanza della regolarizzazione per mitigare il problema della varianza presente nei modelli. Tecniche come il dropout, la batch normalization e la weight decay potrebbero mitigare il fenomeno dell'overfitting, specialmente nei modelli di classificazione delle transazioni, dove la perdita di generalizzazione si manifesta più rapidamente. Inoltre, una gestione più equilibrata delle classi nel dataset, ad esempio tramite strategie di data augmentation, potrebbe migliorare ulteriormente la stabilità e le performance del modello. L'integrazione di queste metodologie rappresenta quindi una direzione promettente per rendere i modelli più robusti ed efficaci nell'identificazione delle transazioni illecite.

Bibliografia

[1] *Gatconv — pytorch geometric documentation*. Accessed: 2025-03-20.

[2] *Google colab notebook*. Accessed: 2025-03-20.

[3] *Heterogeneous graphs — pytorch geometric 2.6.0 documentation*. Accessed: 2025-03-01.

[4] *Sageconv — pytorch geometric documentation*. Accessed: 2025-03-20.

[5] *Transformerconv — pytorch geometric documentation*. Accessed: 2025-03-20.

[6] Elliptic, *Elliptic data set*, 2019. Accessed: 2025-02-28.

[7] EllipticPlusPlus, *Ellipticplusplus*, 2023. Accessed: 2025-02-28.

[8] S. Gtz, *Introduction to graph neural networks: An illustrated guide*, 2025. Accessed: 31 March 2025.

[9] Y. Shi, Z. Huang, S. Feng, H. Zhong, W. Wang, and Y. Sun, *Masked label prediction: Unified message passing model for semi-supervised classification*, 2021.

[10] Y. Shi, Z. Huang, W. Wang, H. Zhong, S. Feng, and Y. Sun, *Masked label prediction: Unified massage passing model for semi-supervised classification*, CoRR, abs/2009.03509 (2020).

[11] G. Su, H. Wang, Y. Zhang, W. Zhang, and X. Lin, *Simple and deep graph attention networks*, Knowledge-Based Systems, 293 (2024), p. 111649.