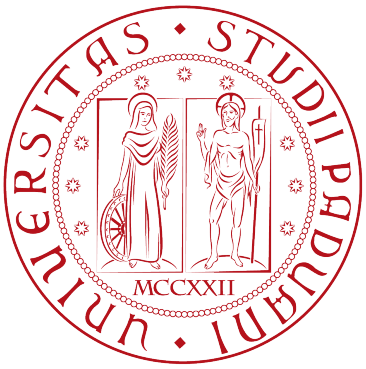


Università degli Studi di Padova

Dipartimento di Ingegneria dell'Informazione

Laurea triennale in Ingegneria Informatica

Titolo

Rilevamento di anomalie del traffico di rete basato su tecniche di elaborazione delle immagini

Supervisore:
Prof.ssa Federica Battisti

Candidato:
Mirko Mannari
1216359

Anno accademico 2022-2023

Abstract

Il progetto ISeeYoo mira a sviluppare un sistema di riconoscimento di anomalie del traffico di rete all'avanguardia tramite rappresentazione bidimensionale di quest ultimo. L'obiettivo di questa tesi è quello di trovare una rappresentazione ottima che permetta di ottenere buoni risultati. Le immagini infatti vengono utilizzate per addestrare una rete neurale convoluzionale la cui architettura verrà approfondita successivamente. L'idea inoltre è quella di dimostrare come, nonostante la fase di addestramento di una rete neurale ricopra un ruolo molto importante, la fase iniziale di elaborazione dei dati di ingresso e di scelta del dataset sia fondamentale. Inoltre, nel caso di questo studio, si vuole mostrare come una diversa rappresentazione del traffico di rete conduca a risultati molto diversi tra loro.

Indice

Abstract **III**

1 Introduzione **1**

1.1 Stato dell'arte 3

2 Related Works **7**

2.1 Dati di input 7

2.2 Modifiche all'architettura della CNN 9

2.3 Addestramento con rappresentazione sparsa 11

3 Proposed Method **15**

3.1 Addestramento con rappresentazione connessa 18

4 Experimental Results **21**

4.1 Test con immagini anomale ideali 23

4.1.1 Test attacco Scan11 23

4.1.2 Test attacco Scan44 24

4.1.3 Test attacco DoS11 25

4.1.4 Test attacco DoS53 26

4.1.5 Test su tutti gli attacchi 27

4.2 Test con immagini anomale ideali e non ideali 28

4.2.1 Test attacco Scan11 29

4.2.2 Test attacco Scan44 30

4.2.3 Test attacco DoS11 31

4.2.4 Test attacco DoS53 32

4.2.5 Test su tutti gli attacchi 33

5 Conclusions **35**

Bibliography **37**

Acknowledgements **39**

Elenco delle figure

1.1 Immagini con rappresentazione sparsa 6

2.1 Infrastruttura di rete del dataset UGR'16 8
2.2 Variational Autoencoder 9
2.3 Autoencoder 9
2.4 Grafico di confronto tra diverse funzioni di attivazione. 10
2.5 Immagini di background con pixel attivi sparsi 12
2.6 Immagini di background con gruppi di pixel attivi 12
2.7 Train\loss delle prime 8 epoche di training con rappresentazione sparsa . . . 13
2.8 Val\loss delle prime 8 epoche di training con rappresentazione sparsa 13

3.1 Grafico della distribuzione di pixel attivi nelle immagini di training 15
3.2 Immagine anomala classificata come immagine di background con rappresentazione sparsa 16
3.3 Esempio di immagini di background con le due rappresentazioni 17
3.4 Esempio di immagini anomale con le due rappresentazioni 17
3.5 Errori di ricostruzione di *training* e *validation* nelle prime 10 epoche di addestramento con rappresentazione connessa 18
3.6 Errori di ricostruzione di *training* e *validation* nelle prime 10 epoche di addestramento con rappresentazione connessa, BCE pesata e *batch normalization* . 19
3.7 Input e output della rete con immagini di background e rappresentazione connessa 20
3.8 Input e output della rete con immagini anomale e rappresentazione connessa . 20

4.1 Grafico risultati di test su immagini ideali con attacco Scan11 24
4.2 Grafico risultati di test su immagini ideali con attacco Scan44 25
4.3 Grafico risultati di test su immagini ideali con attacco DoS11 26
4.4 Grafico risultati di test su immagini ideali con attacco DoS53 27

4.5 Grafico risultati di test su immagini ideali con tutti gli attacchi 28
4.6 Grafico risultati di test su immagini ideali + non ideali con attacco Scan11 . . . 30
4.7 Grafico risultati di test su immagini ideali + non ideali con attacco Scan44 . . . 31
4.8 Grafico risultati di test su immagini ideali + non ideali con attacco DoS11 . . . 32
4.9 Grafico risultati di test su immagini ideali + non ideali con attacco DoS53 . . . 33
4.10 Grafico risultati di test su immagini ideali + non ideali con tutti gli attacchi . . 34

Elenco delle tabelle

4.1 Risultati di test su immagini ideali e attacco Scan11 23
4.2 Risultati di test su immagini ideali e attacco Scan44 25
4.3 Risultati di test su immagini ideali e attacco DoS11 26
4.4 Risultati di test su immagini ideali e attacco DoS53 27
4.5 Risultati di test su immagini ideali con tutti gli attacchi 28
4.6 Risultati di test su immagini ideali + non ideali con attacco Scan11 29
4.7 Risultati di test su immagini ideali + non ideali con attacco Scan44 30
4.8 Risultati di test su immagini ideali + non ideali con attacco DoS11 31
4.9 Risultati di test su immagini ideali + non ideali con attacco DoS53 32
4.10 Risultati di test su immagini ideali + non ideali con tutti gli attacchi 34

Capitolo 1

Introduzione

Il mondo delle telecomunicazioni è ormai divenuto parte integrante delle nostre vite; Internet ricopre un ruolo fondamentale nella società e viene utilizzato in molti ambiti e settori. La digitalizzazione ha subìto un'importante crescita negli ultimi anni portando però con sè anche diverse criticità che è importante non sottovalutare. Una delle tematiche più importanti è sicuramente la sicurezza digitale e quindi l'insieme di sistemi e procedure che permettono ad ogni utente di utilizzare gli strumenti più moderni prevenendo eventuali rischi e problematiche. Purtroppo questo non è sempre possibile ed è in generale difficile garantire con assoluta certezza la sicurezza online: le reti di telecomunicazioni infatti possono essere distinte tra loro e con complessità differenti così come intrusioni e attacchi malevoli che, essendo di varia tipologia ed in continuo aggiornamento, rendono davvero difficile creare un sistema generico in grado di individuarli in tempo reale.

L'idea alla base del progetto ISEEYOO, nato nel 2021, si basa proprio sullo sviluppo di un sistema in grado di rilevare in modo efficace la presenza di una violazione di sicurezza su una rete di connessione. Le strategie per la rilevazione degli attacchi possono dividersi in due categorie: "*rule based detection*", quindi metodi basati sulle firme degli attacchi conosciuti che vengono confrontate con il comportamento osservato sulla rete, e "*anomaly detection*", sistemi generici di rilevamento di anomalie presenti nel traffico di rete osservato [7]. Il secondo metodo, costruendo un profilo di base del traffico di rete normale e rilevando eventuali deviazioni rispetto ad esso, ha il vantaggio di non essere vincolato alle firme dei singoli attacchi; questo gli permette di essere in grado di individuare sia attacchi conosciuti, sia nuovi tipi di violazioni. Lo scopo dello studio è infatti quello di sviluppare un sistema automatico di riconoscimento di generiche anomalie, in particolare utilizzando tecnologie che siano in grado di "imparare" a riconoscere la presenza di attacchi nel traffico di rete.

Diventa quindi importante introdurre il concetto di rete neurale artificiale (*ANN*, *"Artificial Neural Network"*), un sistema di elaborazione computazionale fortemente ispirato al funzionamento dei sistemi nervosi biologici (come il cervello umano). Le *ANN* sono principalmente composte da un elevato numero di nodi computazionali interconnessi (chiamati neuroni), i quali lavorano in modo intrecciato in maniera distribuita per imparare collettivamente dall'input al fine di ottimizzare il risultato finale. Per questo studio tuttavia è stata utilizzata una rete neurale convoluzionale (*CNN*, *"Convolutional Neural Network"*) che, come spiegato in [10], risulta analoga alle tradizionali *ANN* ma ottimizzata in modo da adattarsi al meglio alle specifiche esigenze richieste dal riconoscimento di pattern nelle immagini. I neuroni all'interno delle *CNN* sono infatti organizzati in tre dimensioni: altezza, larghezza (dimensione spaziale dell'input) e profondità.

In particolare per questo studio è stata scelta una precisa architettura che prende il nome di "*Autoencoder*", un tipo specifico di rete neurale progettata principalmente per codificare l'input (nel nostro caso le immagini del traffico di rete) in una rappresentazione compressa e significativa, per poi decodificarla in modo tale che le immagini ricostruite siano il più simile possibile alle originali [1].

L'idea presentata in questa tesi è di trovare nuovi metodi per rappresentare sotto forma di immagini il traffico di rete. Una volta elaborati quindi i dati di input così da ottenere la rappresentazione voluta ed aver creato il dataset, questo è stato utilizzato per addestrare la rete neurale convoluzionale. I risultati ottenuti con le diverse rappresentazioni sono infine stati confrontati per verificare se la nuova rappresentazione da me proposta portasse ad un effettivo miglioramento delle prestazioni del sistema di riconoscimento.

Lo studio è stato strutturato nel seguente modo: dopo una prima fase di introduzione, nella Sezione 1.1 vengono presentate le tecniche più utilizzate per il rilevamento delle anomalie e la loro relativa evoluzione. Successivamente si mostra brevemente l'architettura precedente ("*Variational Autoencoder*") e le modifiche da me apportate ad essa per ottenere la nuova architettura utilizzata per questa tesi ("*Autoencoder*"). Nel capitolo 3 si approfondiscono alcune criticità della rappresentazione sparsa e si presenta quindi la nuova versione "connessa". Infine, nel capitolo 4, si confrontano i risultati ottenuti con le due rappresentazioni e si verifica se il nuovo metodo presentato in questa tesi porti ad effettivi e significativi miglioramenti delle prestazioni della rete.

1.1 Stato dell'arte

La veloce evoluzione di Internet e dei relativi sistemi di comunicazione, come detto in precedenza, ha provocato anche un conseguente e rapido sviluppo di sistemi in grado di esplorare, identificare e sfruttare vulnerabilità informatiche senza autorizzazione. Trovare dei metodi per bloccare o rilevare la presenza di violazioni di questo tipo è diventato quindi un tema molto importante ed è necessario dedicare una sezione di questo studio all'analisi dei sistemi al giorno d'oggi più utilizzati e della loro evoluzione.

I metodi tradizionali per il rilevamento delle anomalie si basano sull'Analisi Statistica Multivariata (*MSA*, *"Multivariate Statistical Analysis"*) [5], un campo della statistica che si occupa dell'analisi e dell'elaborazione simultanea di più variabili, consentendo di identificare relazioni e correlazioni tra di esse. Questo ha permesso di sviluppare sistemi in grado di rilevare pattern e di conseguenza anomalie nell'insieme dei dati, ma lavorare contemporaneamente con una grande quantità di variabili rischia di aumentare notevolmente la complessità computazionale. É stato quindi fondamentale introdurre la *PCA* ("*Principal Component Analysis*") [9], una tecnica di analisi statistica multivariata che, tramite avanzati principi matematici, trasforma un certo numero di variabili possibilmente correlate tra loro in un numero inferiore di variabili chiamate "componenti principali". Riducendo la dimensione dei dati e semplificando quindi l'analisi di essi, si sono registrati significativi miglioramenti delle prestazioni nei sistemi di rilevamento delle anomalie.

Lo studio *"Diagnosing Network-Wide Traffic Anomalies"* [6] propone ad esempio un metodo generale per l'individuazione delle intrusioni: lo spazio ad alta dimensione, occupato dall'insieme di misurazioni del traffico di rete, viene separato in sottospazi disgiunti corrispondenti alle condizioni di rete normali e anomale. L'articolo dimostra infatti come questa divisione possa essere effettuata in modo efficace proprio tramite l'Analisi delle Componenti Principali (*PCA*).

L'utilizzo dell'analisi multivariata per il rilevamento delle anomalie nei dati unita alle tecniche descritte precedentemente è tipicamente indicato come Controllo Statistico dei Processi Multivariato (*MSPC*, *"Multivariate Statistical Process Control"*), il quale utilizza come tecnica principale la *PCA* e ne risolve al contempo alcune criticitá: *MSPC* infatti non si limita solo alla riduzione della dimensionalità dei dati, ma si concentra sulla creazione di modelli statistici multivariati basati sul normale comportamento di essi.
L'articolo [2] mette tuttavia in evidenza come l'introduzione di *PCA* specificatamente nell'analisi di anomalie del traffico di rete presenti delle problematiche da non sottovalutare, in

particolare:

1. Il tasso di falsi positivi è molto sensibile a piccole differenze nel numero di componenti principali presenti nel sottospazio normale.

2. L'efficacia della PCA è sensibile alle diverse aggregazioni delle misurazioni del traffico.

3. Una grande anomalia potrebbe contaminare il sottospazio normale e non essere di conseguenza rilevata.

Viene quindi introdotto il Monitoraggio di Rete Statistico Multivariato (*MSNM*, "*Multivariate Statistical Network Monitoring*"), un approccio statistico che segue la teoria di *MSPC* ma che risulta essere specificamente rivolto al monitoraggio delle reti informatiche, al rilevamento delle anomalie nel traffico di rete e tiene quindi conto delle peculiarità delle reti, come la dinamicità dei flussi di dati. Questo approccio utilizza modelli statistici multivariati per analizzare il traffico di rete, identificare pattern anomali e rilevare eventuali attività sospette o comportamenti non autorizzati.

Oltre alle tecniche basate su *PCA* sono stati sviluppati approcci differenti per rilevare le anomalie del traffico di rete: l'articolo [4] spiega ad esempio l'utilizzo di *ARMA* ("*AutoRegressive Moving Average*"), un modello statistico utilizzato per la previsione di serie temporali. Quest'ultimo permette di catturare le dipendenze lineari tra i valori passati e gli errori residui, consentendo di prevedere il valore futuro di una serie temporale e quindi, nel nostro caso, la presenza di un'intrusione sulla rete. Due ulteriori metodi proposti da [12] per il rilevamento in particolare di attacchi DDoS sono:

1. Metrica di Entropia Generalizzata ("*Generalized Entropy Metric*") che valuta l'incertezza e la variabilità dei dati per rilevare la presenza di un'eventuale violazione.

2. Metrica di Distanza dell'Informazione ("*Information Distance Metric*") la quale confronta la distribuzione statistica del traffico osservato con una distribuzione di riferimento o normale e ne valuta un'eventuale discrepanza.

Il rapido sviluppo delle *ANN* descritte precedentemente ha portato all'introduzione di architetture come *Autoencoder* e *Variational Autoencoder* nella ricerca di nuovi metodi per il rilevamento delle anomalie. Nasce a questo punto l'idea di dare ai dati del traffico di rete una vera e propria rappresentazione grafica, come propone lo studio [11]. Il metodo utilizzato, che prende il nome di grafo di identificazione comportamentale (*BIG*, "*Behavioral Identification Graph*"), si basa infatti su un grafo delle proprietà per rappresentare le associazioni dettagliate

tra le cosìddette ”proprietà comportamentali” dei dati di ingresso. Quest'ultime si riferiscono a caratteristiche e attributi che identificano il traffico di rete come ad esempio indirizzi IP, protocolli di comunicazione, numero di pacchetti, ecc. Questa tecnica permette di ottenere un'unica rappresentazione che racchiude però informazioni più precise sulle interazioni tra le diverse entità nel traffico di rete permettendo così di rilevare eventuali anomalie con maggiore accuratezza.

Ecco quindi che si arriva all'idea alla base del progetto ISEEYOO [3] di rappresentare il traffico di rete tramite immagini 2D che sono state poi utilizzate per addestrare una rete neurale convoluzionale, in particolare un *Variational Autoencoder*, a riconoscere immagini anomale rispetto a quelle ottenute sulla base del traffico normale. La rappresentazione è stata strutturata nel seguente modo:

1. inizialmente è stata creata la variabile $\sum(i,j)$ definita come il numero totale di byte scambiati in una precisa finestra temporale tra un IP destinatario *j* e un IP mittente *i*.

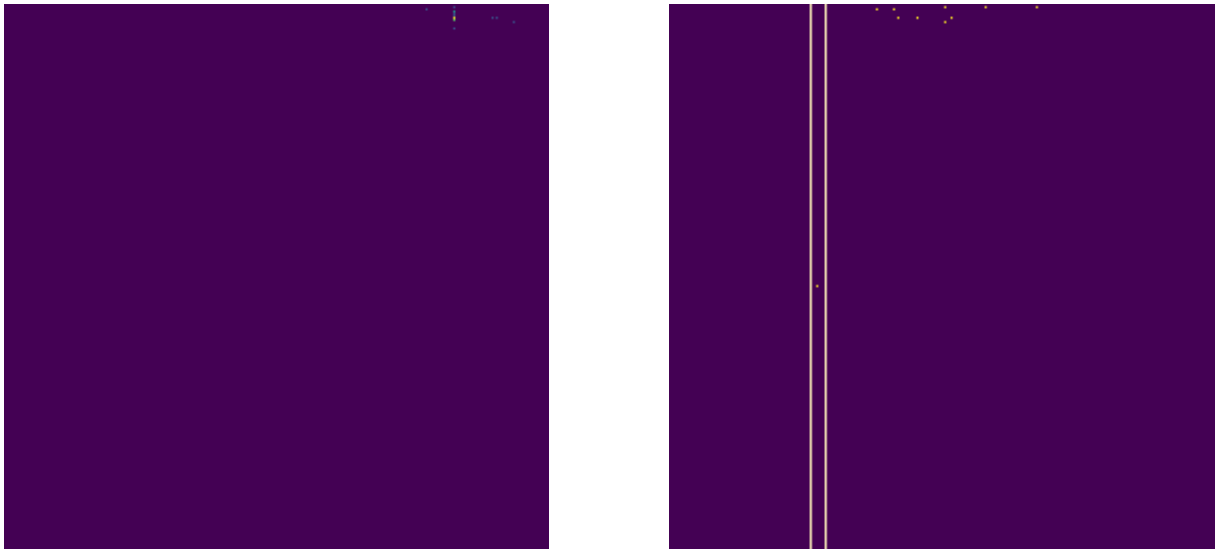
2. successivamente è stato calcolato il valore medio del numero di byte scambiati nell'arco di tempo stabilito, $\mu_{i,j}$, ed è stato sottratto alla variabile precedente ottenendo $\sum_{\mu}(i,j) = \sum(i,j) - \mu_{i,j}$

3. è stata poi calcolata la deviazione standard dei byte scambiati, $\sigma_{i,j}$. La variabile $\sum_{\mu}(i,j)$ è stata divisa per $\sigma_{i,j}$ più un termine molto piccolo per evitare un'eventuale divisione per zero. Il risultato è la variabile $\sum_{\mu,\sigma}(i,j) = \sum_{\mu}(i,j) \,/\, (\sigma_{i,j} + 10^{-4})$

4. infine è stato calcolato il numero di flussi scambiati tra i due nodi durante la finestra temporale e la funzione esponenziale del valore ottenuto, $f_{i,j}$, è stata moltiplicata per la variabile $\sum_{\mu,\sigma}(i,j)$. Il risultato è quindi $\sum^{e}_{\mu,\sigma}(i,j) = \sum_{\mu,\sigma}(i,j) \cdot e^{f_{i,j}}$

La rappresentazione finale, che all'interno di questo studio verrà definita come ”rappresentazione sparsa”, ha subito infine un'ulteriore modifica: mentre le colonne *j* sono rimaste definite dagli indirizzi IP destinatari monitorati, le righe *i* sono state calcolate come l'istogramma della variabile $\sum^{e}_{\mu,\sigma}(i,j)$ precedentemente ottenuta. Il risultato consiste quindi in immagini 256x256 pixel capaci di rappresentare la quantità di flussi scambiati con 256 indirizzi IP destinatari monitorati in una certa finestra temporale, nel mio caso di 1 secondo. La posizione verticale dei pixel attivi identifica o meno la presenza di un'eventuale intrusione.

Di seguito due immagini di esempio (Figura 1.1) che mostrano come, nel caso di presenza di un'anomalia, siano visibili pixel attivi nella zona inferiore dell'immagine (come quello evidenziato all'interno delle due linee gialle); il traffico di rete normale è invece identificato da immagini con pixel attivi presenti esclusivamente nella zona superiore.



(a) Immagine di background

(b) Immagine anomala

Figura 1.1: Immagini con rappresentazione sparsa

Capitolo 2

Related Works

2.1 Dati di input

Prima di analizzare a fondo il modello di rete neurale scelto e spiegare come si sia svolta la fase di addestramento, credo sia importante dedicare un po' di tempo all'analisi della struttura del dataset utilizzato "*UGR'16*" [8]. Quest'ultimo contiene informazioni reali relative al traffico di rete di un fornitore di servizi Internet spagnolo. I dati sono stati raccolti tramite diversi "*netflow collectors*", apparecchiature in grado di intercettare le informazioni provenienti da dispositivi di rete. Il dataset è composto da due differenti set di dati organizzati in settimane: il "*calibration set*" contiene 4 mesi di dati reali raccolti da marzo a giugno 2016 ed è relativo a traffico di rete privo di anomalie e attacchi. La seconda parte del dataset invece, il "*test set*", è formato da dati reali di *background* raccolti da luglio ad agosto 2016 e dati generati artificialmente relativi a traffico di rete anomalo con la presenza di alcuni degli attacchi più conosciuti.

L'infrastruttura della rete monitorata è rappresentata nella Figura 2.1 mentre, per quanto riguarda gli attacchi, sono state prese in considerazione le seguenti classi:

- DoS:
  - DoS11: attacco DoS uno-a-uno dove l'aggressore A_1 attacca la vittima V_{21}.
  - DoS53: 5 aggressori ($A_1 - A_5$) attaccano 3 vittime. In particolare gli aggressori A_1 e A_2 attaccano la vittima V_{21}, A_3 e A_4 attaccano la vittima V_{31} e infine l'aggressore A_5 attacca la vittima V_{41}.
- Port Scanning:
  - Scan11: attacco Scan uno-a-uno dove l'aggressore A_1 attacca la vittima V_{41}.

– Scan44: attacco di scansione in cui gli aggressori A_1, A_2, A_3 e A_4 scansionano rispettivamente le vittime V_{21}, V_{11}, V_{31} e V_{41}.

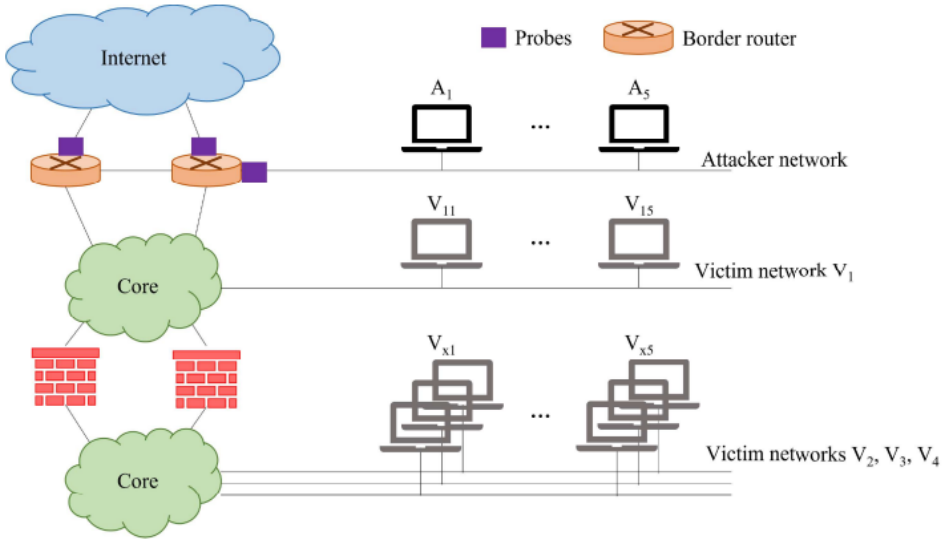


Figura 2.1: Infrastruttura di rete del dataset UGR'16

Per questo studio, i dati anomali di test sono stati separati in due gruppi distinti, denominati rispettivamente *ideali* e *non ideali*. Nel primo gruppo sono inclusi solo i dati delle immagini in cui tra l'indirizzo IP del mittente e l'indirizzo IP del destinatario sono stati scambiati più di 35 flussi di rete, mentre nel secondo gruppo sono presenti i dati delle immagini con un numero di flussi di rete inferiore a 35. Il modello dopo la fase di addestramento è stato testato su entrambi i gruppi, prima su dati non ideali e successivamente su dati ideali per verificare come questa distinzione influenzi le prestazioni della rete neurale.

Prima di iniziare effettivamente ad addestrare la rete neurale, il dataset ha subito un'ulteriore modifica: dalle immagini di *training* è stato infatti selezionato e rimosso un campione pari al 10% del totale che è poi stato utilizzato per la fase di *validation* durante la quale viene calcolato l'errore di ricostruzione medio della rete. Il valore ottenuto durante quest'ultimo processo verrà utilizzato successivamente per classificare le immagini durante la fase di *test*. In questo modo ho ottenuto due *split* distinti di immagini: uno di *training* composto da 288000 immagini e uno di *validation* composto da 32000 immagini. Questi due *set* devono sempre rimanere distinti tra loro ed è fondamentale garantire che rimangano i medesimi durante tutta la durata dell'addestramento per evitare quello che viene definito fenomeno di "*Data Leakage*", problematica che si verifica quando informazioni esterne al dataset di *training* vengono utilizzate per addestrare il modello. Questi dati aggiuntivi consentirebbero al modello di imparare o elaborare informazioni che non dovrebbe conoscere invalidando così le reali performance.

Nel nostro caso, se delle immagini del *validation set* venissero usate durante il *training*, la rete andrebbe ad apprendere informazioni anche da quest'ultime, diminuendo così l'errore di ricostruzione medio commesso durante la fase di *validation* e invalidando di conseguenza la classificazione delle immagini.

2.2 Modifiche all'architettura della CNN

Una delle fasi iniziali di questa tesi è stata l'analisi dell'architettura della rete neurale convoluzionale: il modello precedente risultava infatti molto avanzato e di conseguenza forse eccessivamente complesso. Le immagini venivano infatti fornite in input ad una rete neurale convoluzionale (CNN) che veniva addestrata al riconoscimento delle anomalie. L'architettura del modello precedente era di tipo "*variational autoencoder*" (Figura 2.2) addestrato utilizzando come *loss function* $L = BCE + KLD$ e quindi la somma tra "*Binary Cross Entropy*" e "*Kullback–Leibler Divergence*"

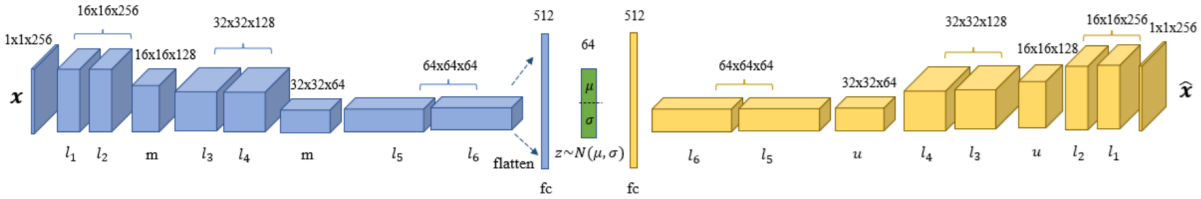


Figura 2.2: Variational Autoencoder

Il primo passo è stato quindi quello di semplificarlo: ho utilizzato un *autoencoder* (Figura 2.3), un modello di rete neurale convoluzionale differente in cui il numero di layer convoluzionali è stato ridotto a 3 ed il *latent space* portato a 20 *features*.

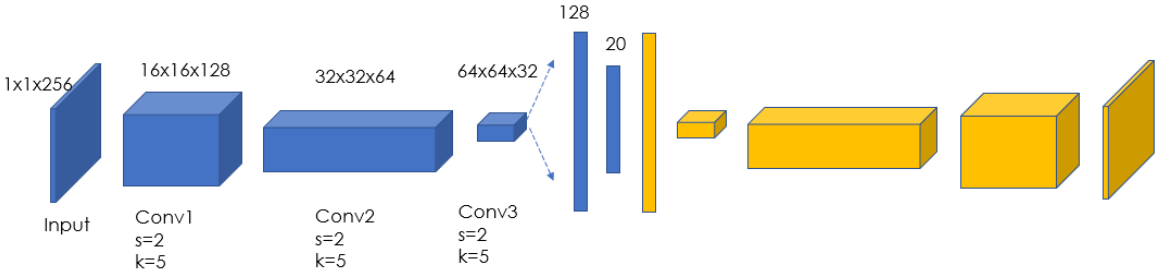


Figura 2.3: Autoencoder

Come funzione di attivazione è stata utilizzata la *ELU* ("*Exponential Linear Unit*"), un'alternativa della tradizionale *ReLU* ("*Rectified Linear Unit*") che però ne corregge alcune criticità.

$$ELU\,(x) = \begin{cases} x & \text{if x} > 0 \\ \alpha\,(e^x - 1) & \text{if x} \leq 0 \end{cases} \tag{2.1}$$

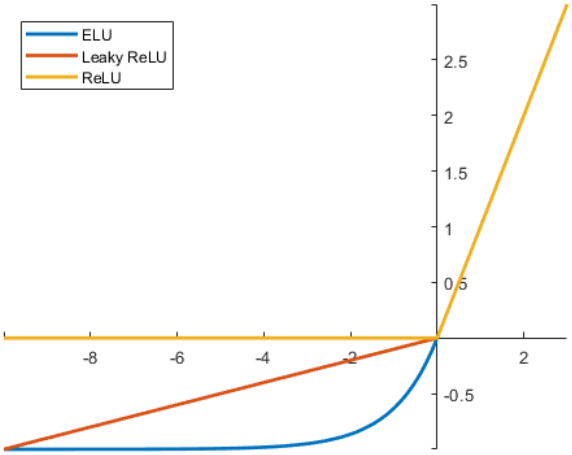


Figura 2.4: Grafico di confronto tra diverse funzioni di attivazione.

Asse x: input del neurone. Asse y: output della funzione di attivazione.

Come si può notare dal grafico precedente (Figura 2.4), la funzione *ELU* permette valori negativi: questo risolve la cosìddetta "*Dying ReLU*" cioè quando la somma pesata dei segnali in ingresso ad un neurone è negativa, provocando un output nullo. Questa condizione porta ad un rallentamento del *training* poiché il neurone, avendo gradiente nullo, non fornisce più contributo all'apprendimento della rete. Queste modifiche hanno reso notevolmente più facile l'intero lavoro e allo stesso tempo hanno permesso di ottenere ottimi risultati come vedremo nelle sezioni successive.

L'effettivo funzionamento e utilizzo della rete mostrata sopra risulta essere il seguente: durante la fase di *validation*, al termine di ogni epoca di *training*, il modello calcola l'errore di ricostruzione su immagini diverse da quelle utilizzate per la fase di addestramento più precisamente appartenenti al *validation set* come spiegato sopra. Durante questa fase, gli errori vengono salvati ed utilizzati per calcolare un valore di soglia o *threshold* che viene infine utilizzato durante la fase di test. Quando viene fornita un'immagine in ingresso alla rete, quest'ultima la ricostruisce calcolando l'errore commesso; se il valore risulta essere maggiore della *threshold* ottenuta precedentemente, l'immagine di input viene catalogata come anomala.

Per aumentare le prestazioni della rete e migliorare quindi i risultati ottenuti dall'analisi delle metriche è stata introdotta una *pfa* ("*Probability of False Alarm*"), una misura statisti-

ca che rappresenta la probabilità che il modello classifichi un'immagine di background come anomala. L'introduzione di questa probabilità permette di ridurre il valore di soglia riducendo di conseguenza la precisione del modello in quanto ci saranno più immagini prive di attacco che verranno classificate come anomale. Accettando questo compromesso, si ottiene però un notevole incremento della metrica *Recall* in quanto sarà molto più probabile che l'errore di ricostruzione di un'immagine con anomalia sia maggiore del valore di soglia e che quindi tale immagine sia correttamente classificata come anomala. Tutti i test sono stati infatti effettuati con 4 differenti probabilità: 0, 0.01, 0.05 e 0.1. Come vedremo successivamente i due valori che hanno permesso di ottenere risultati migliori sono 0.01% e 0.05%.

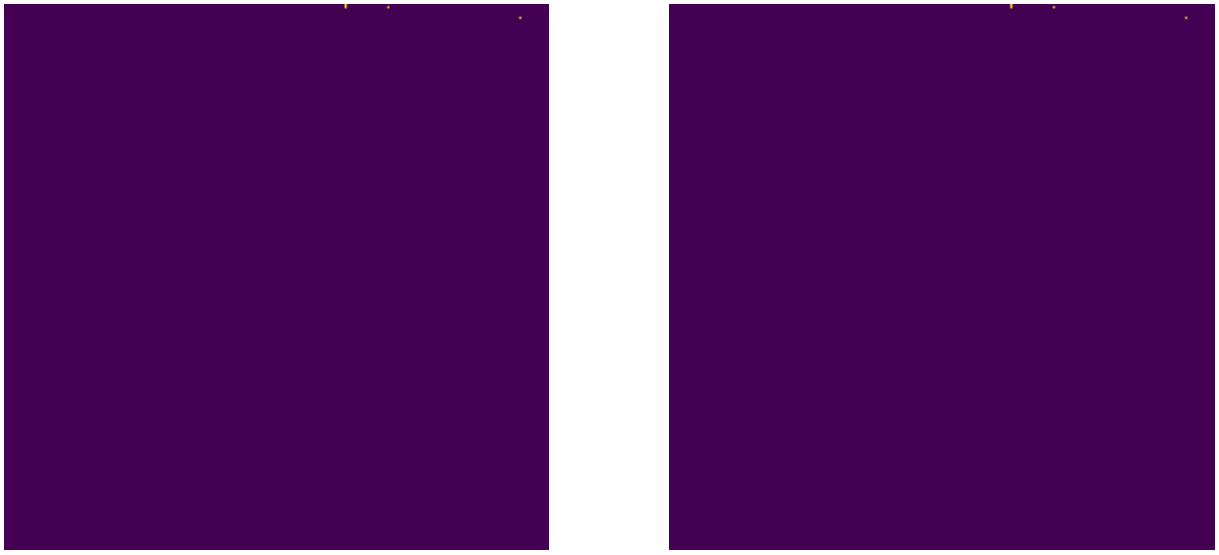
2.3 Addestramento con rappresentazione sparsa

Inizialmente la rete è stata addestrata su 320000 immagini con rappresentazione sparsa di cui il 90% (288000) per l'effettiva fase di *training* e il restante 10% (32000) per la fase di *validation*. Per cominciare è stata scelta come *loss function* la *BCE* ("*Binary Cross Entropy*") che ha però presentato criticità fin da subito a causa della minima quantità di pixel attivi nelle immagini in relazione ai pixel a zero; dopo pochi step di addestramento infatti la rete ricostruiva le immagini ponendo a zero ogni pixel. In questo modo l'errore commesso dal modello risultava essere troppo piccolo per permettere alla stessa di imparare o cercare nuove soluzioni.

Per risolvere il problema è stata implementata una diversa *loss function*, in particolare una *BCE* pesata che, come dice il termine, va a dare un peso maggiore ai pixel attivi aumentando notevolmente il valore dell'errore di ricostruzione commesso dal modello ogni qualvolta questo metta un pixel originariamente attivo (valore 1) a zero. Per questo studio il peso applicato ai pixel attivi è pari a $\omega = 15$ e quindi l'errore commesso su di essi è 15 volte maggiore di quello commesso sui pixel a zero. Questa correzione alla *loss function* ha permesso alla rete di addestrarsi correttamente iniziando dopo diverse epoche a ricostruire bene le immagini di *training* e riducendo notevolmente ad ogni epoca l'errore di ricostruzione commesso durante fase di *validation*. Tuttavia il modello faceva ancora fatica a ricostruire immagini che presentavano punti molto vicini tra loro; questo problema si è sicuramente ridotto all'avanzare delle epoche di addestramento ma non è purtroppo del tutto sparito.

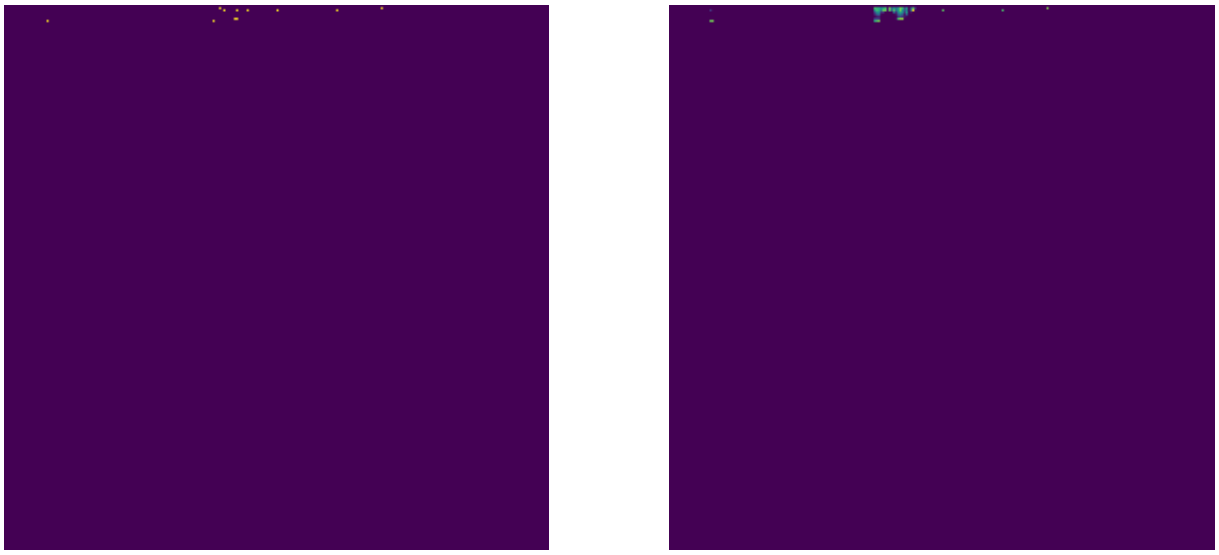
In Figura 2.5 e 2.6 alcune immagini raccolte durante la fase di *training* che mostrano il confronto tra l'immagine originale fornita in input alla rete e quella ricostruita dalla stessa. Come si nota, nel caso in cui i pixel attivi risultano sparsi o comunque abbastanza distanti tra loro (Figura 2.5), la rete riesce a ricostruire bene l'immagine distinguendo correttamente i differenti

punti. Nel secondo caso invece (Figura 2.6), quando sono presenti gruppi di pixel attivi, cioè zone in cui sono presenti pixel con valore diverso da zero relativamente vicini tra loro, la rete fatica a ricostruire correttamente l'immagine e non riesce a distinguere i singoli punti.



(a) Immagine di background (b) Immagine di background ricostruita

Figura 2.5: Immagini di background con pixel attivi sparsi



(a) Immagine di background (b) Immagine di background ricostruita

Figura 2.6: Immagini di background con gruppi di pixel attivi

Il modello, con questo tipo di rappresentazione, è stato addestrato in circa 23 ore utilizzando una scheda video Nvidia GeForce RTX 3060 per un totale di 23 epoche ottenendo come valori medi degli errori di ricostruzione nelle fasi di *training* e *validation* rispettivamente 0.000041 e 0.000048. L'introduzione della *batch normalization* tra i layer convoluzionali, come anticipato nella sezione precedente, ha reso l'addestramento più stabile e veloce come si può notare

dai grafici "*Train\loss*" (Figura 2.7) e "*Val\loss*" (Figura 2.8) successivi che mostrano l'andamento dell'errore di ricostruzione medio rispettivamente nella fase di *training* e nella fase di *validation*.

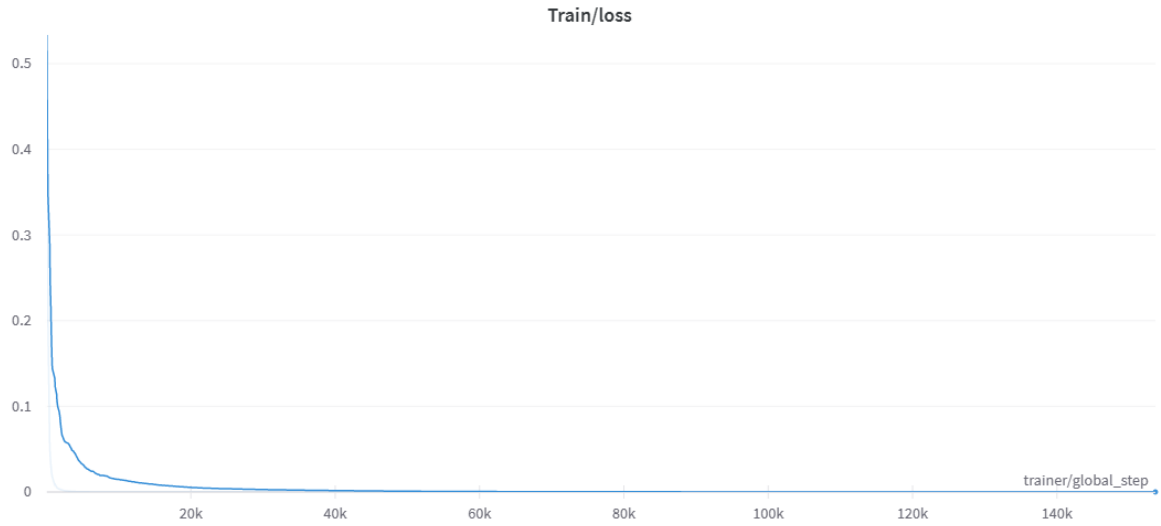


Figura 2.7: Train\loss delle prime 8 epoche di training con rappresentazione sparsa

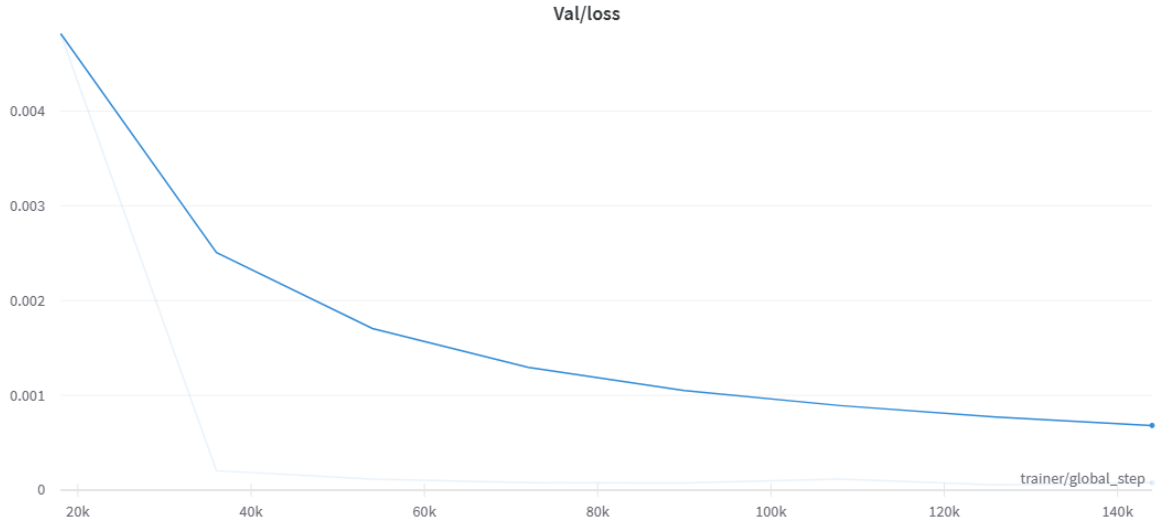


Figura 2.8: Val\loss delle prime 8 epoche di training con rappresentazione sparsa

Capitolo 3

Proposed Method

La rappresentazione sparsa risultava essere inizialmente buona ed ha permesso di iniziare a sviluppare un sistema di riconoscimento delle anomalie basato non più solo sui dati numerici del traffico di rete, ma su vere e proprie immagini generate in funzione dei dati stessi. Tuttavia, come vedremo meglio nella sezione dei risultati, questa rappresentazione presenta alcuni problemi: il modello ha infatti dimostrato nelle fasi di test di fare molta fatica a riconoscere un pattern efficace che permetta di distinguere correttamente le immagini di background da quelle anomale, dando spesso più importanza alla posizione orizzontale dei pixel attivi (indirizzo IP del destinatario) piuttosto che quella verticale (funzione del numero di flow scambiati tra mittente e destinatario). Uno dei problemi principali è infatti la distribuzione di pixel attivi nelle immagini di *training*: come si nota nel grafico sottostante (Figura 3.1), i dati di *training* presentano un maggiore scambio di informazioni con indirizzi IP di valore elevato.

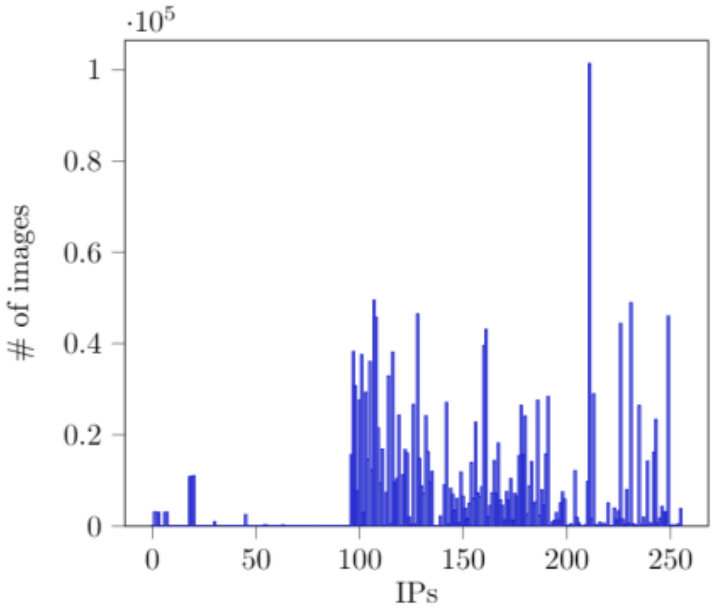
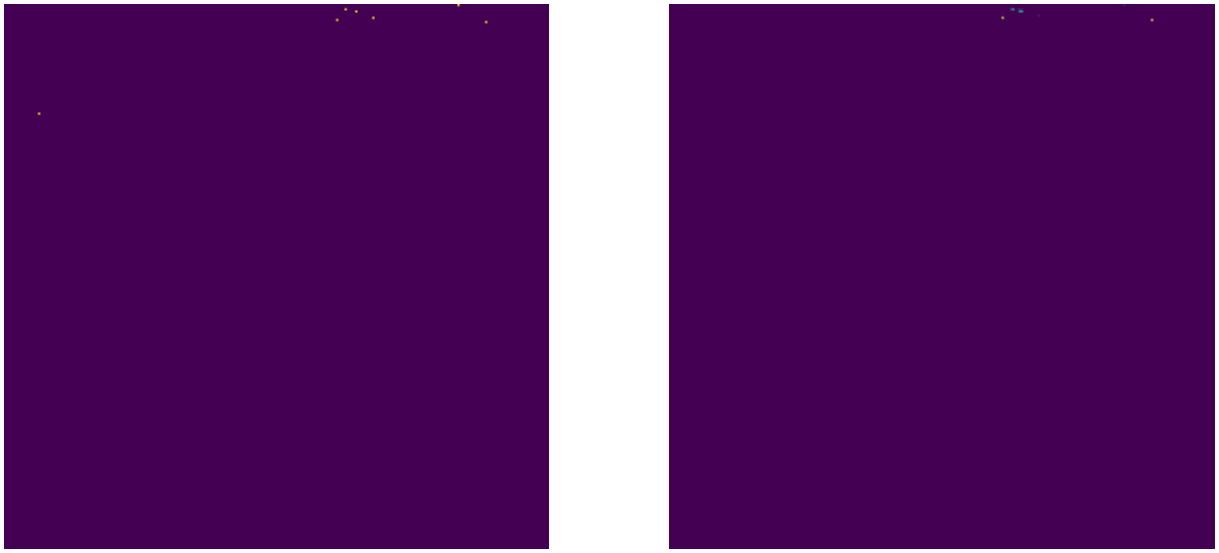


Figura 3.1: Grafico della distribuzione di pixel attivi nelle immagini di training

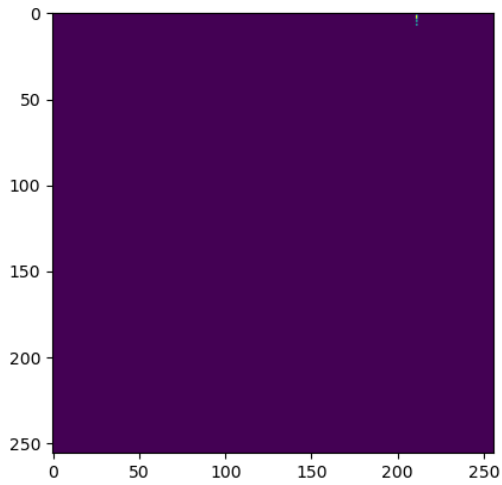
Nelle immagini impiegate per l'addestramento della rete, si osserva una maggiore presenza di pixel attivi nella regione centrale/destra. Di conseguenza, vi è il rischio che la rete faccia un uso improprio di tale caratteristica, incontrando difficoltà nel riconoscere e ricostruire immagini non anomale con pixel attivi nella regione sinistra. Ne deriva un modello che, anziché classificare le immagini in base alla posizione verticale dei pixel, lo fa tenendo erroneamente conto anche della posizione orizzontale. Di seguito un esempio (Figura 3.2) in cui il modello non riesce a rilevare la presenza di anomalie nell'immagine e, ricostruendola con un errore minimo, la classifica di conseguenza come di *background*.



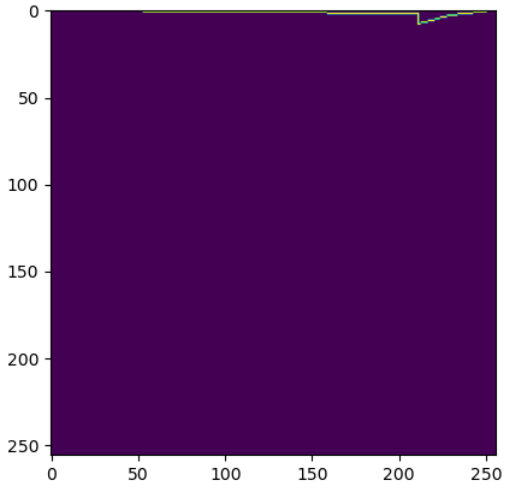
(a) Immagine anomala (b) Immagine anomala ricostruita

Figura 3.2: Immagine anomala classificata come immagine di background con rappresentazione sparsa

Nasce quindi la necessità di aumentare l'informazione visiva delle immagini così da permettere al modello di distinguere in modo più chiaro le immagini di *background* da quelle anomale. L'idea è quella di creare una rappresentazione che metta meglio in evidenza i pixel attivi e ne valorizzi la posizione verticale: è stata quindi creata una rappresentazione "connessa" in cui ogni punto con valore maggiore di zero viene collegato con una linea ai suoi due pixel attivi orizzontalmente più vicini. In questo modo, anche a primo impatto visivo, risulta essere più evidente la differenza tra immagini di *background*, che sono ora caratterizzate da una linea più o meno dritta nella regione superiore, e quelle anomale, che presentano invece picchi verso la regione inferiore.

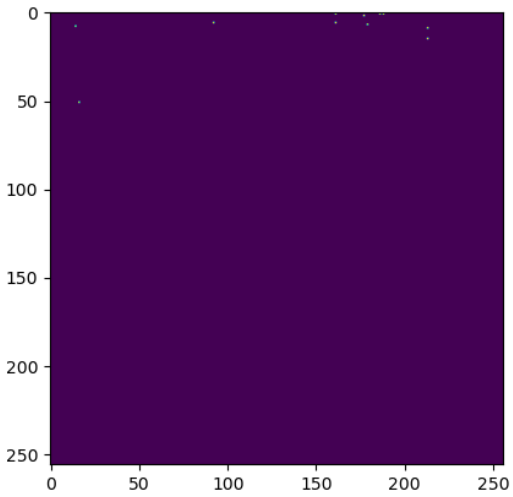


(a) Rappresentazione sparsa

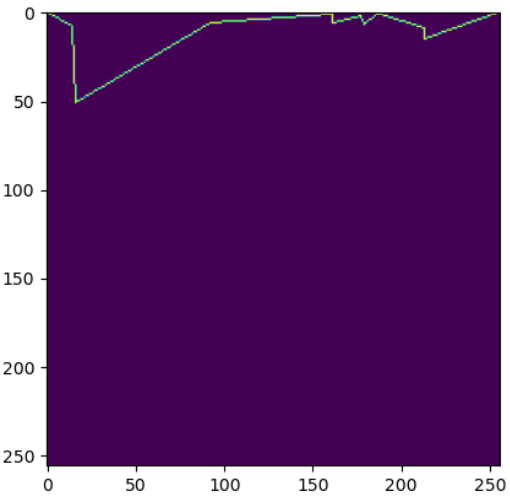


(b) Rappresentazione connessa

Figura 3.3: Esempio di immagini di background con le due rappresentazioni



(a) Rappresentazione sparsa



(b) Rappresentazione connessa

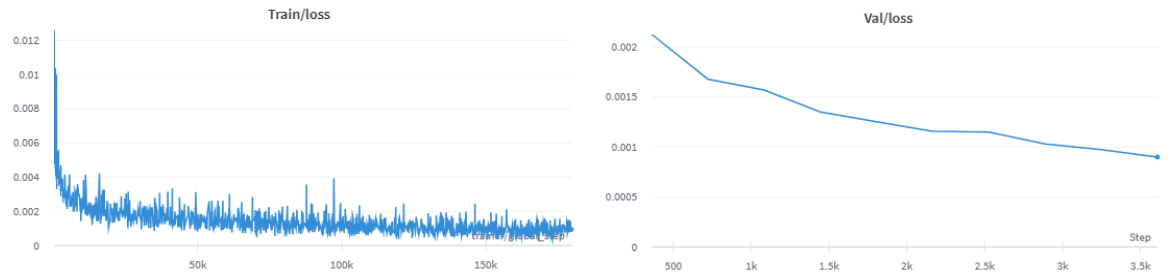
Figura 3.4: Esempio di immagini anomale con le due rappresentazioni

Come si può notare nelle figure precedenti (Figura 3.3 e Figura 3.4), le immagini ora risultano essere più chiare e distinguibili in modo sicuramente più evidente. L'introduzione di questa nuova rappresentazione infatti ha favorito un notevole incremento delle prestazioni della rete, permettendo quindi di rilevare con più facilità l'eventuale presenza di anomalie nel traffico di rete. Con questo nuovo approccio, il modello riesce ad identificare in modo più efficace i pattern caratteristici delle immagini di traffico normale, valorizzando maggiormente la posizione verticale dei pixel attivi. Di conseguenza, le immagini di background verranno ricostruite correttamente durante la fase di addestramento. Fornendo in ingresso immagini anomale, la rete ricostruirà l'input con un errore significativamente maggiore, aumentando così la probabilità di una corretta classificazione.

3.1 Addestramento con rappresentazione connessa

Anche in questo caso il dataset di ingresso è stato diviso in due *split*: il 10% del totale di immagini di addestramento è stato infatti rimosso ed è stato utilizzato per la fase di *validation*, per un totale di 288000 immagini per il *training set* e 32000 per il *validation set*. Per garantire un corretto confronto tra i risultati ottenuti tramite l'approccio basato sulla rappresentazione sparsa e quelli derivanti dal nuovo metodo proposto, per l'addestramento sono state impiegate le stesse immagini utilizzate per l'approccio basato sulla rappresentazione sparsa. Tuttavia, queste sono state elaborate così da ottenerne la nuova versione connessa sulla quale è stata poi addestrata la rete.

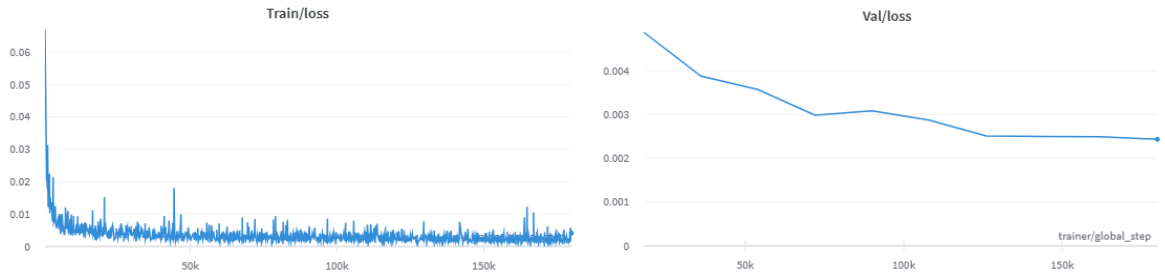
Il modello, con questo nuovo metodo, è stato inizialmente addestrato utilizzando BCE ("*Binary Cross Entropy*") come *loss function* per un totale di 25 epoche. Al termine dell'addestramento, si sono ottenuti errori di ricostruzione medi di 0.0001934 nella fase di *training* e 0.0005947 nella fase di *validation*. Quelli ottenuti, come vedremo meglio nel Capitolo 4, si rivelano già essere ottimi risultati: la rete infatti, con la nuova rappresentazione connessa, è riuscita fin da subito a ricostruire correttamente le immagini, senza l'ausilio della BCE pesata utilizzata invece per la precedente rappresentazione sparsa. La nuova rappresentazione proposta aumenta notevolmente la quantità di informazione sulla quale la rete neurale si addestra. Questo permette quindi di evitare la problematica riscontrata con la precedente rappresentazione sparsa per la quale, usando la semplice BCE e non la sua versione pesata, dopo poche epoche di addestramento la rete ricostruiva le immagini ponendo erroneamente ogni pixel a zero.



(a) Train\Loss (b) Val\Loss

Figura 3.5: Errori di ricostruzione di *training* e *validation* nelle prime 10 epoche di addestramento con rappresentazione connessa

Una volta terminato quindi il *training* e raccolti i risultati della fase di *test* che verranno approfonditi nel capitolo successivo, la rete è stata addestrata nuovamente. In questa occasione è stata però introdotta la *batch normalization* tra i layer convoluzionali, così da stabilizzare e rendere più veloce l'apprendimento, ed è stata implementata anche in questo caso la BCE pesata già utilizzata in precedenza per la rappresentazione sparsa. L'intento era quello di verificare se, dando un peso maggiore ai pixel attivi anche nelle immagini connesse, ci fosse ancora margine di miglioramento e se le prestazioni della rete sarebbero aumentate. Il peso dato ai pixel attivi risulta essere anche in questo caso $\omega = 15$. Applicate quindi le modifiche, al termine di questo secondo addestramento effettuato sempre su immagini con rappresentazione connessa, gli errori di ricostruzione commessi dalla rete nella fase di *training* e nella fase di *validation* risultano essere rispettivamente 0.002511 e 0.001724.

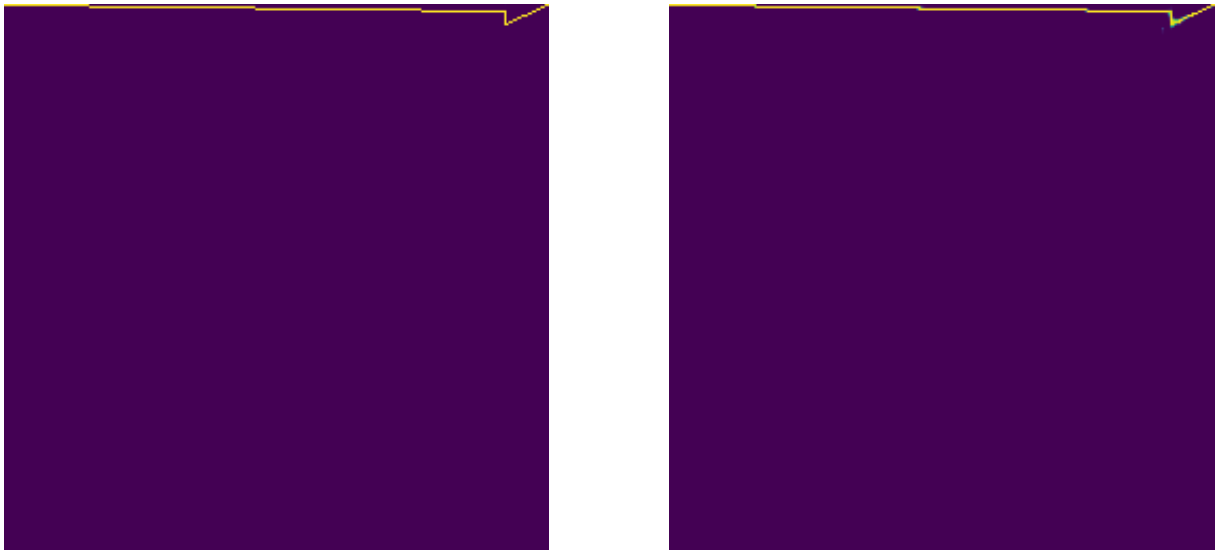


(a) Train\Loss (b) Val\Loss

Figura 3.6: Errori di ricostruzione di *training* e *validation* nelle prime 10 epoche di addestramento con rappresentazione connessa, BCE pesata e *batch normalization*

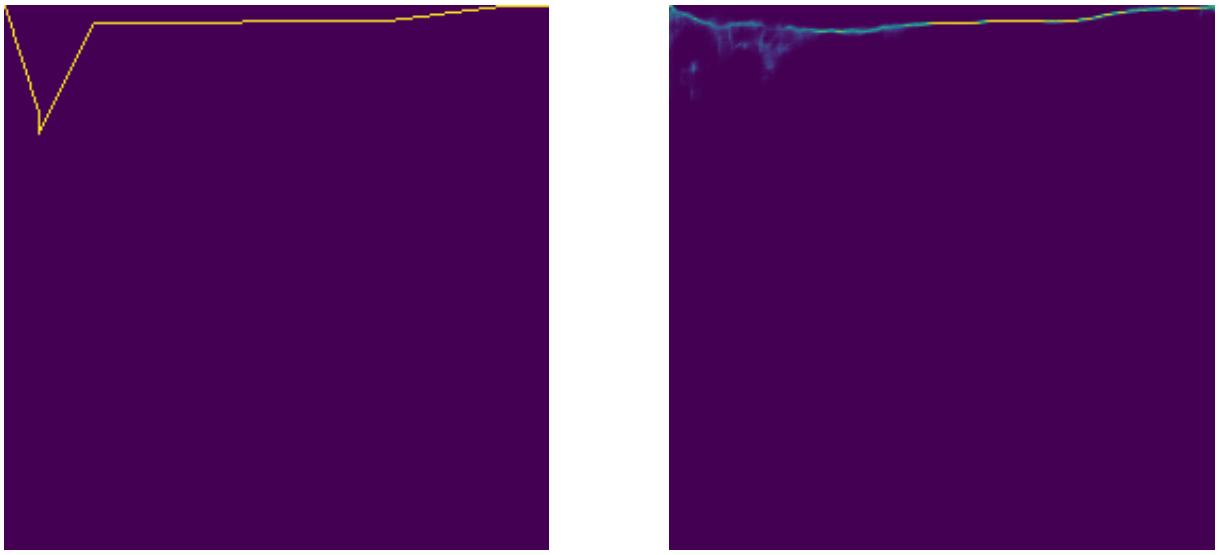
Mettendo a confronto i Grafici 3.5a e 3.6a, la curva di apprendimento nel secondo caso risulta essere più ripida nelle fasi iniziale di *training*. Questo è dovuto all'introduzione della *batch normalization* tra i layer convoluzionali che, oltre a velocizzare l'addestramento, ne permette una maggiore stabilizzazione. Come si può notare infatti, la curva del Grafico 3.6a risulta essere meno "disturbata" ed i picchi dovuti alle variazioni dell'errore risultano limitati rispetto al Grafico 3.5a.

Di seguito vengono presentati degli esempi (Figura 3.7 e Figura 3.8) di immagini fornite in input alla rete e della loro rispettiva ricostruzione. Come si può notare, l'*autoencoder* ricostruisce molto bene le immagini di background mentre in generale tende a fallire la ricostruzione di immagini anomale, commettendo su di esse un errore maggiore.



(a) Immagine di background (b) Immagine di background ricostruita

Figura 3.7: Input e output della rete con immagini di background e rappresentazione connessa



(a) Immagine anomala (b) Immagine anomala ricostruita

Figura 3.8: Input e output della rete con immagini anomale e rappresentazione connessa

Capitolo 4

Experimental Results

Questo capitolo si concentra sull'esporre i risultati sperimentali ottenuti dopo i relativi addestramenti. La fase di *test*, per valutare al meglio le prestazioni della rete sulle diverse rappresentazioni, è stata organizzata secondo la seguente procedura: inizialmente i test sono stati effettuati su un dataset composto da una parte di immagini di background e una parte di immagini anomale "ideali", relative quindi a traffico di rete anomalo in cui, tra mittente e destinatario, è stato scambiato un numero di flussi ≥ 35. Successivamente i test sono stati ripetuti unendo nello stesso dataset sia immagini anomale "ideali", sia immagini anomale "non ideali", le quali sono caratterizzate da un numero di flussi scambiati < 35. Questo ha permesso di ottenere una visione globale delle prestazioni della rete, includendo nella valutazione anche i casi meno ottimali. I due casi vengono trattati separatamente nelle Sezioni 4.1 e 4.2 all'interno delle quali vengono approfonditi i test effettuati in cinque sottosezioni distinte: quattro di queste si concentrano sui test singoli per tipologia di attacco (scan11, scan44, dos11, dos53), l'ultima mostra invece i risultati della rete unendo insieme tutte le immagini "ideali" con ogni tipologia di attacco nello stesso dataset.

Come spiegato nella Sezione 2.2, durante la fase di test è stata introdotta una *pfa* ("*Probability of False Alarm*") che, andando ad agire direttamente sul valore di soglia utilizzato per classificare le immagini, ha permesso di migliorare notevolmente le prestazioni della rete. Dopo diversi tentativi per ricercare il valore ottimale di *pfa* da applicare, i risultati migliori sono stati ottenuti utilizzando 0.01% e 0.05%.

In ciascuna delle sottosezioni interne ai capitoli 4.1 e 4.2 vengono esposti i risultati dei test effettuati sulle due tipologie di rappresentazioni e sulle rispettive architetture. In totale, per ogni sottosezione, vengono considerati tre diversi casi di test ripetuti per i due valori di *pfa* selezionati. Il primo caso si basa su rappresentazione sparsa con autoencoder addestrato

utilizzando BCE pesata e *batch normalization*. Il secondo è invece relativo alla nuova rappresentazione connessa proposta con autoencoder addestrato utilizzando la semplice BCE. Il terzo e ultimo caso riprende la stessa rappresentazione del precedente ma si basa questa volta sull'autoencoder addestrato utilizzando BCE pesata e *batch normalization*.

Per ogni test le metriche prese in considerazione sono le seguenti:

1. "*Accuracy*" definita come la percentuale di classificazioni corrette rispetto al totale di classificazioni effettuate dalla rete.

2. "*Precision*" definita, nel nostro caso in particolare, come la percentuale di immagini anomale correttamente classificate rispetto al totale di immagini classificate come anomale.

3. "*Recall*" definita, nel nostro caso in particolare, come la percentuale di immagini anomale correttamente classificate rispetto al totale di immagini anomale.

4. "*F1Score*", una misura che combina la *precision* e la *recall* in un singolo valore e che tiene quindi conto sia dei falsi positivi (errore di *precisione*) sia dei falsi negativi (errore di *recall*)

A causa di un forte sbilanciamento dei dati nei dataset di test, il numero di immagini di *background* risulta essere molto maggiore di quello delle immagini anomale. La metrica "*Accuracy*" avrà quindi sempre valori molto alti superiori al 90%. Di conseguenza, per valutare le prestazioni della rete, viene data maggior rilevanza alle altre tre metriche sopra citate e quindi "*Precision*" "*Recall*" e "*F1Score*"

Un'osservazione importante riguarda la nomenclatura utilizzata durante i test. Nelle tabelle di dati che verranno presentate nelle sottosezioni successive, si usano i seguenti termini:

1. "*dots*" fa riferimento alla rappresentazione sparsa e quindi all'autoencoder addestrato con BCE pesata e *batch normalization*.

2. "*connected dots*" fa riferimento alla rappresentazione connessa e quindi all'autoencoder addestrato con normale BCE come *loss function*.

3. "*connected dots wBCE*" fa riferimento alla rappresentazione connessa e quindi all'autoencoder addestrato questa volta con BCE pesata (wBCE *weighted Binary Cross Entropy*) e *batch normalization*.

4.1 Test con immagini anomale ideali

In questa prima sezione ci si concentra sui test della rete neurale effettuati su un dataset composto da una parte di immagini di *background* e da una parte di immagini anomale puramente "ideali".

4.1.1 Test attacco Scan11

In questo caso il dataset di test è diviso come segue: 60000 immagini di *background* e 776 immagini anomale ideali (ovvero circa l'1% del dataset) relative a traffico di rete caratterizzato da attacco di tipo "Scan11". Come si nota nella Tabella 4.1, l'incremento delle prestazioni implementando la nuova rappresentazione connessa è notevole. La rete fatica molto a classificare le immagini con pixel sparsi ottenendo valori di "*Recall*" inferiori al 3% e "*Precision*" inferiore al 10-20%. Con rappresentazione connessa invece si ottengono risultati ottimi raggiungendo valori di "*Precision*" e "*Recall*" pari a circa 80% e 60%. Tuttavia, come si nota nel grafico in Figura 4.1, l'applicazione di BCE pesata e *batch normalization* all'addestramento della rete su immagini con rappresentazione connessa, ha permesso di distribuire l'incremento di prestazioni in modo più equo tra le diverse metriche. Il risultato è un importante miglioramento generale della capacità della rete di riconoscere immagini anomale, portando tutte le metriche ad un valore medio pari circa a 80-90%.

		Scan11			
		F1Score	Accuracy	Precision	Recall
PFA 0.05%	dots pfa 0.05%	0.04138	0.9863	0.1915	0.0232
	connected dots pfa 0.05%	0.7447	0.9922	0.8459	0.6651
	connected dots wBCE pfa 0.05%	0.8617	0.9962	0.7987	0.9356
PFA 0.01%	dots pfa 0.01%	0.004981	0.9869	0.07407	0.002577
	connected dots pfa 0.01%	0.7038	0.9915	0.8833	0.5849
	connected dots wBCE pfa 0.01%	0.8783	0.9969	0.8777	0.8789

Tabella 4.1: Risultati di test su immagini ideali e attacco Scan11

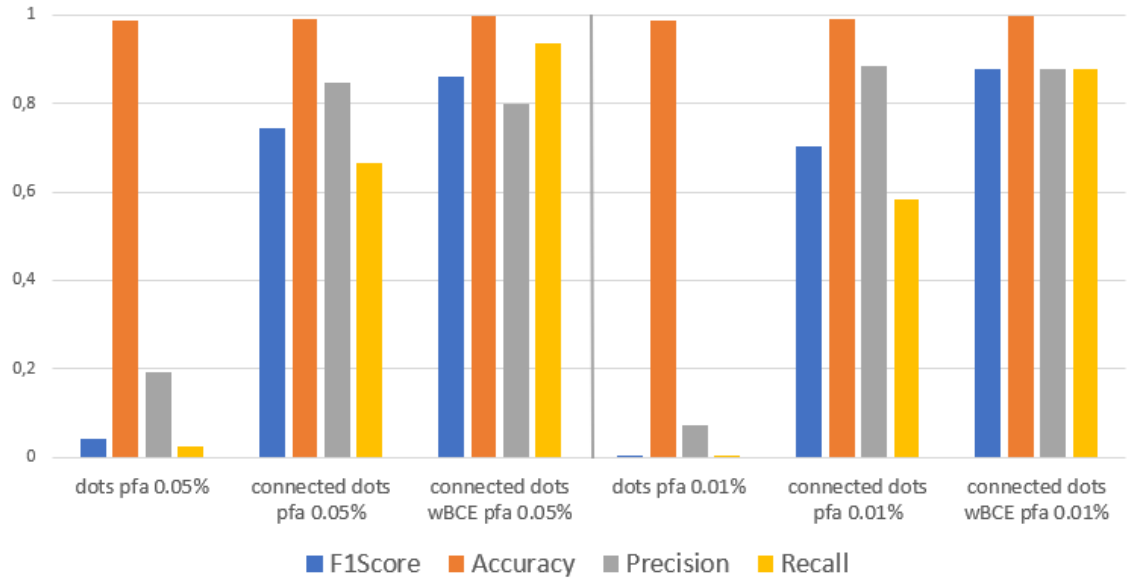


Figura 4.1: Grafico risultati di test su immagini ideali con attacco Scan11

4.1.2 Test attacco Scan44

In questo caso il dataset di test è diviso come segue: 60000 immagini di *background* e 2532 immagini anomale ideali (ovvero il 4% del dataset) relative a traffico di rete caratterizzato da attacco di tipo ”Scan44”. Per i risultati ottenuti dai test su questo tipo di attacco, valgono le stesse considerazioni della Sezione 4.1.1. Anche qui la rete fatica molto a classificare le immagini con rappresentazione sparsa. Si ha un importante incremento delle prestazioni introducendo semplicemente la rappresentazione connessa, ma uno ancora maggiore introducendo BCE pesata e *batch normalization*. Grazie a quest'ultime modifiche, si ottiene una rete neurale capace di distinguere molto bene le immagini anomale con attacco ”Scan44” da quelle di *background*. Come si nota anche dal grafico in Figura 4.2, in questo caso le metriche si aggirano intorno ad un valore ancora maggiore di quello ottenuto nei test su attacco ”Scan11”: si parla infatti di un valor medio pari a circa 95%.

		Scan44			
		F1Score	Accuracy	Precision	Recall
PFA 0.05%	dots pfa 0.05%	0.01974	0.9587	0.2549	0.01027
	connected dots pfa 0.05%	0.9509	0.996	0.9499	0.9518
	connected dots wBCE pfa 0.05%	0.9483	0.9957	0.9304	0.9668
PFA 0.01%	dots pfa 0.01%	0.002344	0.9592	0.1071	0.001185
	connected dots pfa 0.01%	0.9293	0.9945	0.9655	0.8957
	connected dots wBCE pfa 0.01%	0.9471	0.9958	0.9614	0.9333

Tabella 4.2: Risultati di test su immagini ideali e attacco Scan44

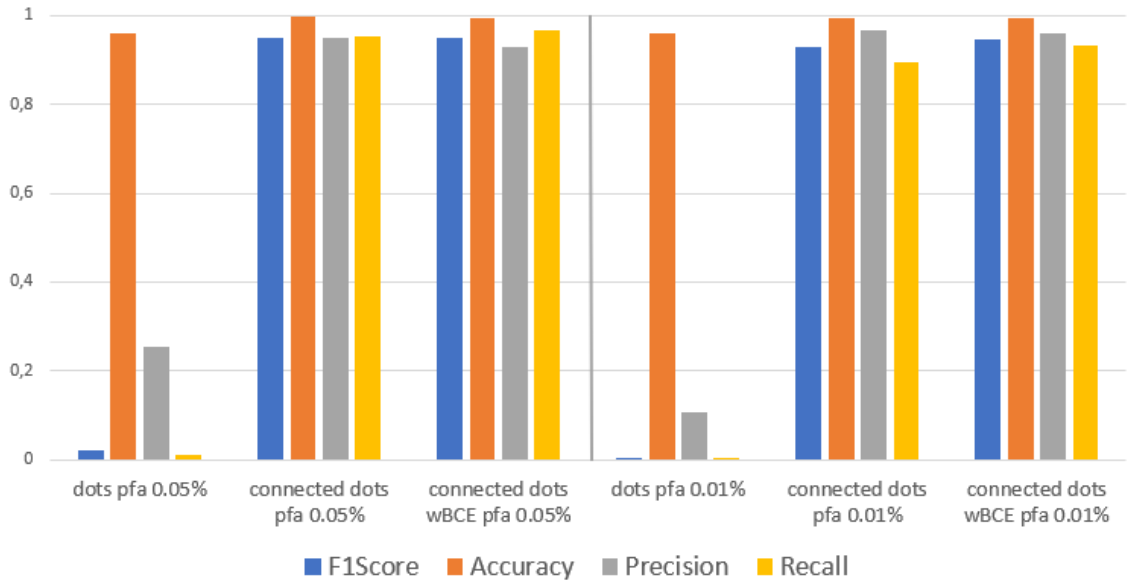


Figura 4.2: Grafico risultati di test su immagini ideali con attacco Scan44

4.1.3 Test attacco DoS11

In questo caso cambia il tipo di attacco che è ora di tipo DoS. Il dataset di test è diviso come segue: 60000 immagini di *background* e 1659 immagini anomale ideali (ovvero circa il 2-3% del dataset) relative a traffico di rete caratterizzato da attacco di tipo ”DoS11”. Come si nota nella Tabella 4.3, nel caso di *pfa* 0.01% il miglioramento delle prestazioni introducendo BCE pesata e *batch normalization* è notevole, soprattutto per quanto riguarda la metrica ”*Recall*”: si passa infatti da un 44% ad un 97%, con un incremento quindi di oltre il 50%. Anche in questo caso la rete performa davvero bene riuscendo a riconoscere la maggior parte delle immagini anomale. Con rappresentazione connessa e BCE pesata, le metriche si aggirano intorno ad un valore medio pari a circa 96%.

		DoS11			
		F1Score	Accuracy	Precision	Recall
PFA 0.05%	dots pfa 0.05%	0.001152	0.9719	0.01299	0.0006028
	connected dots pfa 0.05%	0.9601	0.9978	0.9285	0.994
	connected dots wBCE pfa 0.05%	0.9459	0.9969	0.9003	0.9964
PFA 0.01%	dots pfa 0.01%	0.001187	0.9727	0.03846	0.0006028
	connected dots pfa 0.01%	0.5962	0.9838	0.9012	0.4454
	connected dots wBCE pfa 0.01%	0.9578	0.9977	0.9443	0.9717

Tabella 4.3: Risultati di test su immagini ideali e attacco DoS11

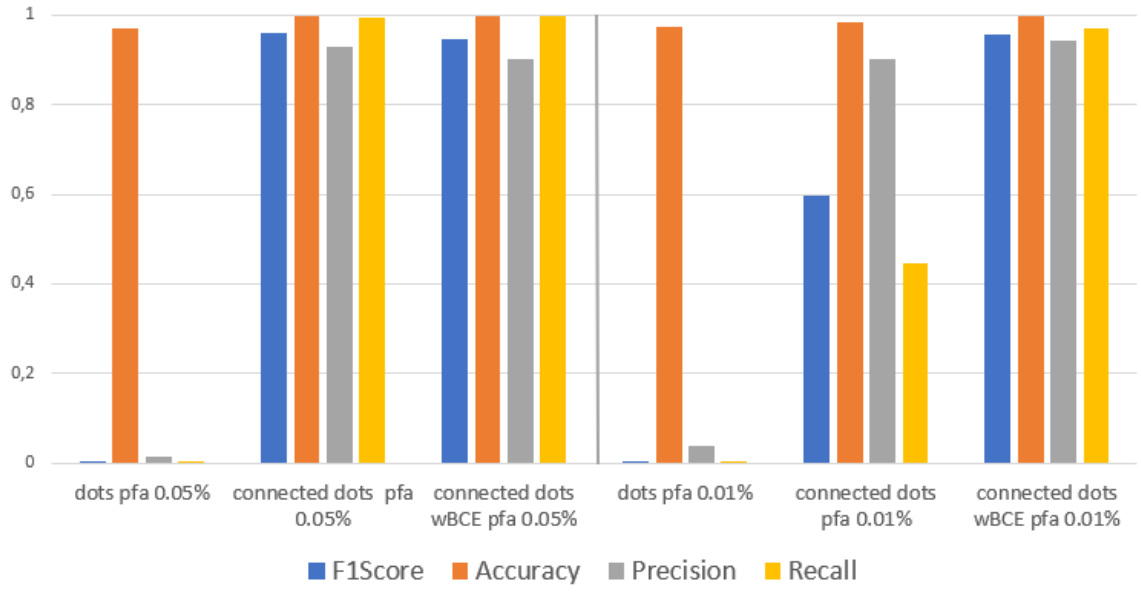


Figura 4.3: Grafico risultati di test su immagini ideali con attacco DoS11

4.1.4 Test attacco DoS53

Il dataset per questo tipo di test è diviso nel seguente modo: 60000 immagini di *background* e 4574 immagini anomale ideali (ovvero circa il 7% del dataset) relative a traffico di rete caratterizzato da attacco di tipo "DoS53". Tuttavia in questo caso, come si può notare dalla Tabella 4.4, le prestazioni della rete risultano essere più scarse. Nel caso migliore, quindi con rappresentazione connessa, BCE pesata e *batch normalization*, si ha infatti una "*Recall*" di circa 50%. La difficoltà della rete nel classificare correttamente immagini con attacco DoS53 è legata al metodo attraverso il quale i dati del traffico di rete vengono tradotti in una rappresentazione 2D. Molte delle immagini anomale del dataset con attacco DoS53 risultano infatti veramente simili ad immagini di *background* e presentano pixel attivi principalmente nella regione supe-

riore. Come si nota anche nel grafico in Figura 4.4, la "*Precision*" mantiene comunque valori elevati superiori al 90%: la rete infatti classifica correttamente la maggior parte delle immagini relative a traffico di rete normale e commette errore solo nella classificazione di immagini anomale.

		DoS53			
		F1Score	Accuracy	Precision	Recall
PFA 0.05%	dots pfa 0.05%	0.02001	0.9287	0.3821	0.01028
	connected dots pfa 0.05%	0.6918	0.9657	0.9514	0.5435
	connected dots wBCE pfa 0.05%	0.7008	0.9661	0.9334	0.561
PFA 0.01%	dots pfa 0.01%	0.004772	0.9289	0.3056	0.002405
	connected dots pfa 0.01%	0.3183	0.9416	0.9158	0.1926
	connected dots wBCE pfa 0.01%	0.6664	0.9638	0.9609	0.5101

Tabella 4.4: Risultati di test su immagini ideali e attacco DoS53

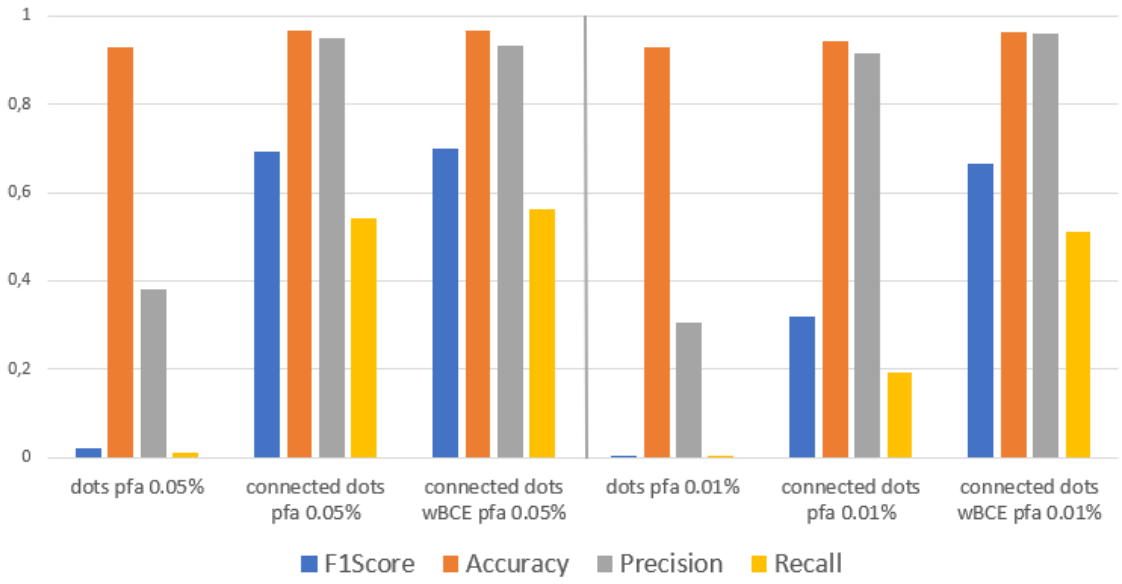


Figura 4.4: Grafico risultati di test su immagini ideali con attacco DoS53

4.1.5 Test su tutti gli attacchi

In questa sezione si trattano i test effettuati sempre su immagini "Ideali" ma unendo nello stesso dataset tutte e quattro le tipologie di attacco. In questo modo è stato possibile ottenere una visione generale sulla capacità della rete di riconoscere immagini anomale, indipendentemente dal tipo di violazione. Il dataset è suddiviso nel seguente modo: 60000 immagini di

background e 9541 immagini anomale ideali (ovvero circa il 13% del dataset) relative a traffico di rete caratterizzato da un generico attacco dei quattro presi in considerazione in questa tesi. Nella Tabella 4.5 e nel grafico in Figura 4.5 si possono notare le prestazioni generali della rete che risultano essere molto buone. Il risultato finale è un modello che, con rappresentazione connessa, classifica correttamente oltre il 70% delle immagini anomale, con una "*Precision*" circa del 97% ed un "*F1Score*" del 85-86%.

		All Attacks			
		F1Score	Accuracy	Precision	Recall
PFA 0.05%	dots pfa 0.05%	0.01895	0.863	0.5476	0.009643
	connected dots pfa 0.05%	0.8565	0.9651	0.9828	0.759
	connected dots wBCE pfa 0.05%	0.8638	0.9665	0.9758	0.7749
PFA 0.01%	dots pfa 0.01%	0.003548	0.8627	0.4048	0.001782
	connected dots pfa 0.01%	0.6374	0.9264	0.9823	0.4718
	connected dots wBCE pfa 0.01%	0.8409	0.962	0.9866	0.7326

Tabella 4.5: Risultati di test su immagini ideali con tutti gli attacchi

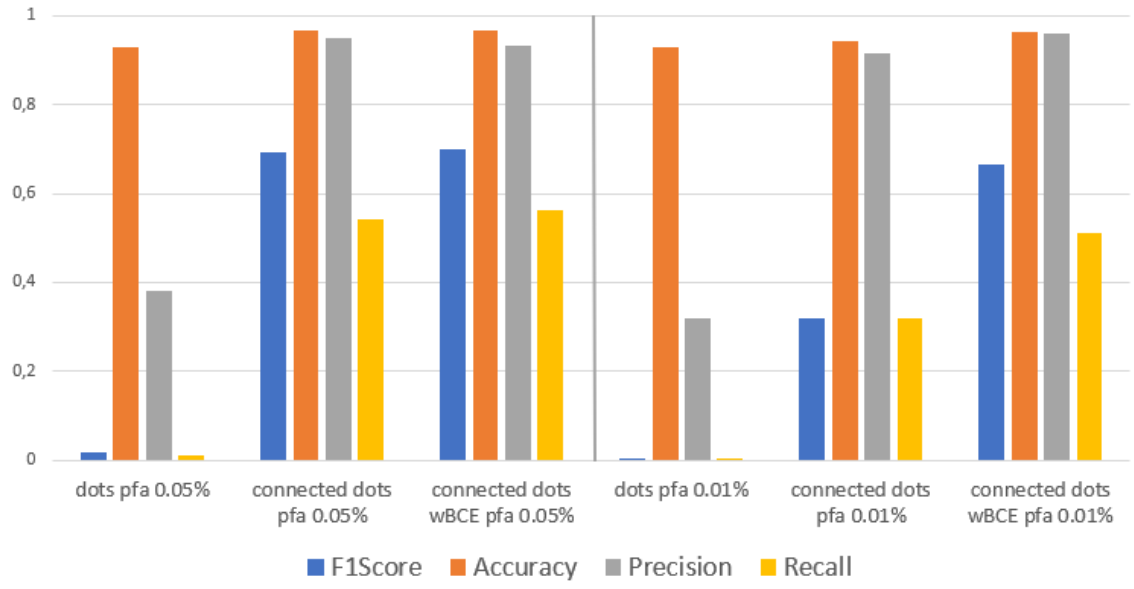


Figura 4.5: Grafico risultati di test su immagini ideali con tutti gli attacchi

4.2 Test con immagini anomale ideali e non ideali

In questa sezione vengono mostrati i risultati dei test effettuati su un dataset composto in parte da immagini di *background* ed in parte da immagini anomale "ideali" e "non ideali" insieme.

L'idea è quella di verificare il comportamento della rete non solo nel caso ottimale ma anche nel caso in cui il traffico di rete anomalo presenti un numero di flussi minore. Avendo introdotto casi meno ottimali, le prestazioni della rete sono in generale calate. Tuttavia, come vedremo in maniera approfondita nelle singole sottosezioni successive, i risultati dei test rimangono comunque buoni.

4.2.1 Test attacco Scan11

Il dataset per questo test è stato diviso nel seguente modo: 60000 immagini di *background* e 992 immagini anomale (ovvero circa l'1-2% del dataset) relative a traffico di rete caratterizzato da attacco di tipo "Scan11". Delle 992 immagini con violazione, 776 sono "ideali" mentre le restanti 216 sono "non ideali". Come si nota in Tabella 4.6, le prestazioni della rete sono sicuramente calate rispetto al caso analizzato nella Sezione 4.1.1. Si parla infatti di circa un 70% di immagini anomale correttamente classificate (77% nel caso migliore con rappresentazione connessa, BCE pesata e *pfa* dello 0.05%), a differenza dei test effettuati solo su immagini "ideali" in cui si era ottenuto un valore di "*Recall*" pari al 93% nel caso migliore. Tuttavia, anche in questo test, introducendo la nuova rappresentazione connessa si ha un notevole incremento delle prestazioni.

		Scan11			
		F1Score	Accuracy	Precision	Recall
PFA 0.05%	dots pfa 0.05%	0.03496	0.9828	0.2	0.01915
	connected dots pfa 0.05%	0.7683	0.9931	0.8461	0.7036
	connected dots wBCE pfa 0.05%	0.7936	0.9934	0.8086	0.7792
PFA 0.01%	dots pfa 0.01%	0.003925	0.9834	0.07407	0.002016
	connected dots pfa 0.01%	0.7272	0.9925	0.8833	0.6179
	connected dots wBCE pfa 0.01%	0.7711	0.9934	0.8777	0.6875

Tabella 4.6: Risultati di test su immagini ideali + non ideali con attacco Scan11

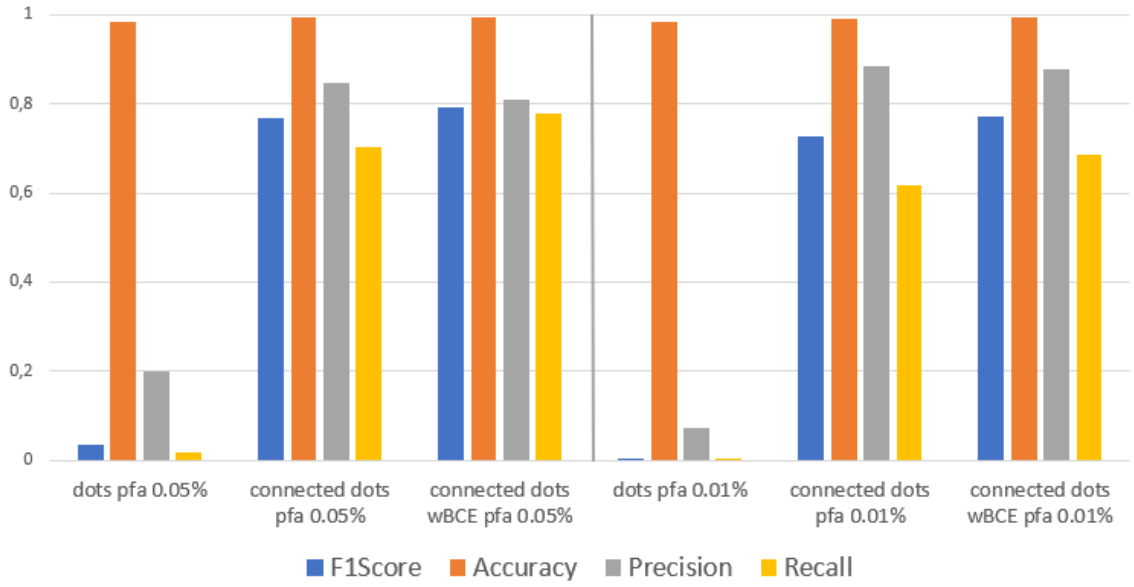


Figura 4.6: Grafico risultati di test su immagini ideali + non ideali con attacco Scan11

4.2.2 Test attacco Scan44

In questo caso il dataset di test è diviso nel seguente modo: 60000 immagini di *background* e 3198 immagini anomale (ovvero circa il 5% del dataset) relative a traffico di rete caratterizzato da attacco di tipo ”Scan44”. Delle 3198 immagini con violazione, 2532 sono ”ideali” mentre le restanti 666 sono ”non ideali”. Anche in questo caso si nota un calo generale delle prestazioni, soprattutto per quanto riguarda la metrica ”*Recall*”; quest'ultima è infatti passata da oltre un 90% misurato nei test della Sezione 4.1.2 ad un 77% nel caso migliore (rappresentazione connessa e BCE pesata), come si nota in Tabella 4.7.

		Scan44			
		F1Score	Accuracy	Precision	Recall
PFA 0.05%	dots pfa 0.05%	0.01696	0.9486	0.2692	0.008755
	connected dots pfa 0.05%	0.8417	0.9856	0.9501	0.7555
	connected dots wBCE pfa 0.05%	0.8484	0.9859	0.9316	0.7789
PFA 0.01%	dots pfa 0.01%	0.00186	0.949	0.1071	0.0009381
	connected dots pfa 0.01%	0.818	0.984	0.9655	0.7095
	connected dots wBCE pfa 0.01%	0.8358	0.9853	0.9614	0.7392

Tabella 4.7: Risultati di test su immagini ideali + non ideali con attacco Scan44

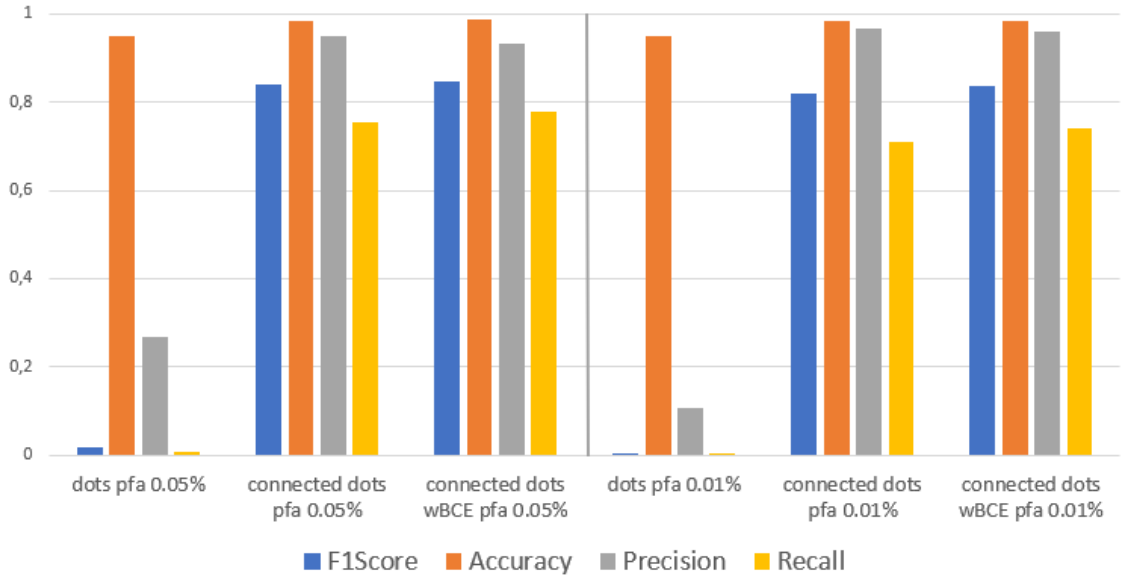


Figura 4.7: Grafico risultati di test su immagini ideali + non ideali con attacco Scan44

4.2.3 Test attacco DoS11

Per questo test il dataset risulta diviso nel seguente modo: 60000 immagini di *background* e 1676 immagini anomale (ovvero circa l'2-3% del dataset) relative a traffico di rete caratterizzato da attacco di tipo "DoS11". Delle 1676 immagini con violazione, 1659 sono "ideali" mentre le restanti 17 sono "non ideali". In questo caso, essendo in numero notevolmente inferiore, le immagini "non ideali" hanno influenzato molto poco i risultati rispetto al test trattato nella Sezione 4.1.3. Come si nota in Tabella 4.8 ed anche nel grafico in Figura 4.8, le prestazioni della rete nel caso migliore (rappresentazione connessa e BCE pesata) sono ottime con valori delle metriche anche di molto superiori al 90%.

		Dos11			
		F1Score	Accuracy	Precision	Recall
PFA 0.05%	dots pfa 0.05%	0.001141	0.9716	0.01299	0.0005967
	connected dots pfa 0.05%	0.9554	0.9975	0.9285	0.9839
	connected dots wBCE pfa 0.05%	0.9413	0.9967	0.9003	0.9863
PFA 0.01%	dots pfa 0.01%	0.001175	0.9724	0.03846	0.0005967
	connected dots pfa 0.01%	0.5921	0.9835	0.9012	0.4409
	connected dots wBCE pfa 0.01%	0.953	0.9974	0.9443	0.9618

Tabella 4.8: Risultati di test su immagini ideali + non ideali con attacco DoS11

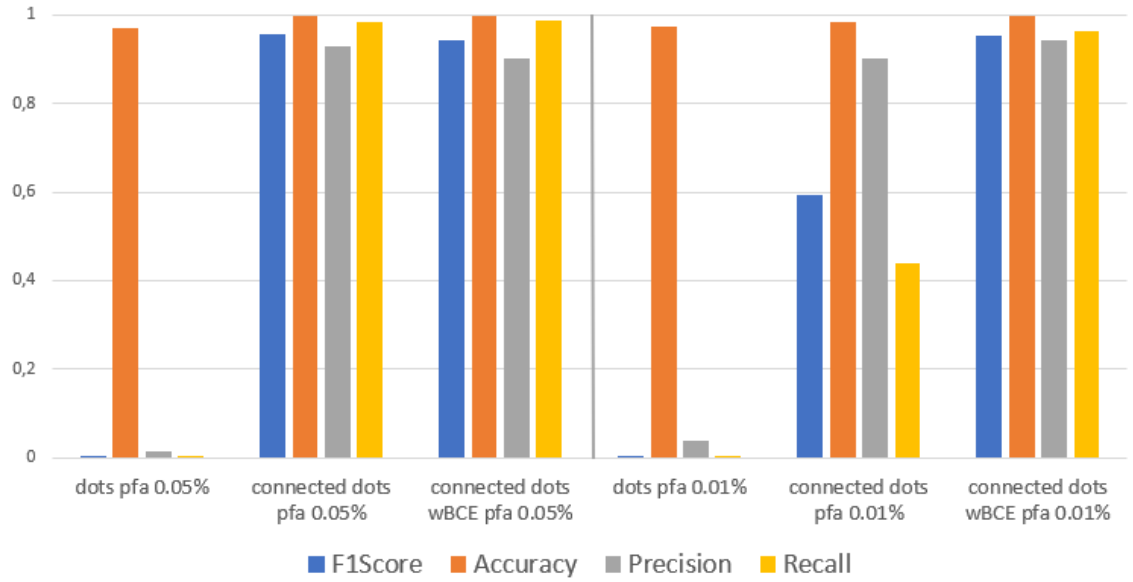


Figura 4.8: Grafico risultati di test su immagini ideali + non ideali con attacco DoS11

4.2.4 Test attacco DoS53

Con quest'ultimo tipo di attacco, il dataset di test è diviso come segue: 60000 immagini di *background* e 4596 immagini anomale (ovvero circa il 7% del dataset) relative a traffico di rete caratterizzato da attacco di tipo "DoS53". Delle 4596 immagini con violazione, 4574 sono "ideali" mentre le restanti 22 sono "non ideali". Come nel caso presentato nella Sezione 4.1.4, la rete fatica molto a classificare correttamente immagini anomale con attacco DoS53. Nel caso migliore infatti, quindi con rappresentazione connessa e BCE pesata, il modello riconosce correttamente solo il 55% delle immagini anomale. L'errore è commesso solo su immagini con violazioni e non su quelle di *background*: questo è confermato dai valori di "*Precision*" che, nel caso di rappresentazione connessa, non scendono mai sotto al 90%, come si nota in Tabella 4.9.

		Dos53			
		F1Score	Accuracy	Precision	Recall
PFA 0.05%	dots pfa 0.05%	0.01992	0.9284	0.3821	0.01023
	connected dots pfa 0.05%	0.6899	0.9654	0.9514	0.5411
	connected dots wBCE pfa 0.05%	0.6989	0.9658	0.9335	0.5585
PFA 0.01%	dots pfa 0.01%	0.00475	0.9286	0.3056	0.002393
	connected dots pfa 0.01%	0.317	0.9412	0.9158	0.1917
	connected dots wBCE pfa 0.01%	0.6643	0.9635	0.9609	0.5076

Tabella 4.9: Risultati di test su immagini ideali + non ideali con attacco DoS53

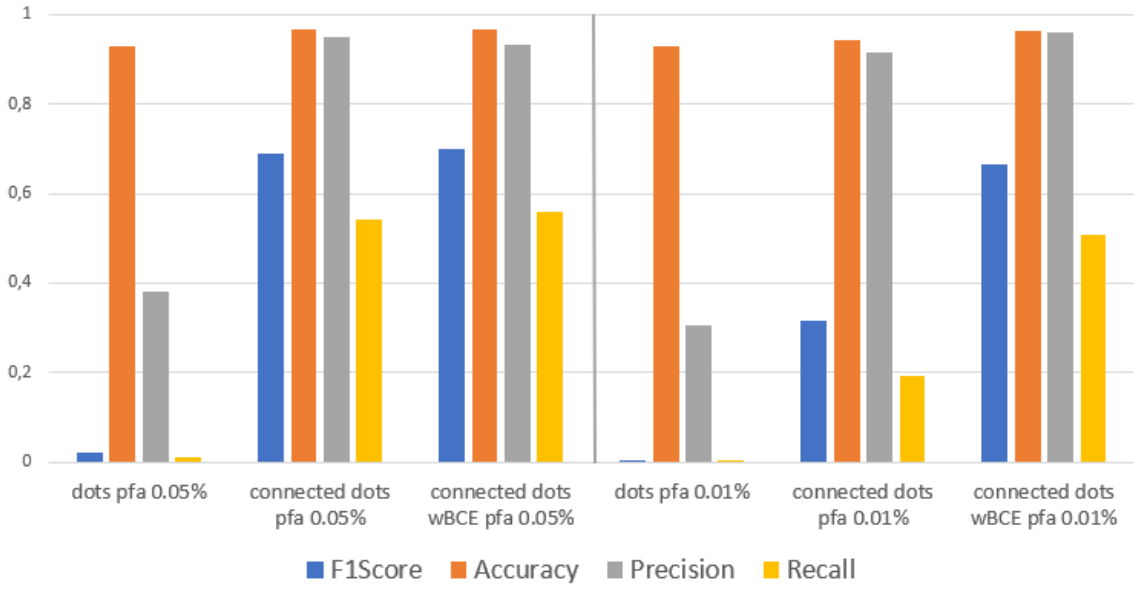


Figura 4.9: Grafico risultati di test su immagini ideali + non ideali con attacco DoS53

4.2.5 Test su tutti gli attacchi

In quest'ultimo caso di test si vuole verificare la capacità della rete di classificare correttamente una generica immagine anomala, indipendentemente dal tipo di attacco e, questa volta, anche dal numero di flussi e quindi indipendentemente dalla sua idealità. Il dataset è diviso come segue: 60000 immagini di *background* e 10462 immagini anomale (ovvero circa il 15% del dataset) relative a traffico di rete caratterizzato da un generico attacco dei quattro presi in considerazione per questa tesi. Delle 10462 immagini con violazione, 9541 sono "ideali" mentre le restanti 921 sono "non ideali". La Tabella 4.10 ed il grafico in Figura 4.10 mettono in evidenza i risultati finali ottenuti: la rete performa generalmente bene, con una capacità di classificare correttamente circa il 70% delle immagini anomale. Il caso migliore si conferma essere quello con rappresentazione connessa, BCE pesata e valore di *pfa* pari allo 0.05%. La precision rimane molto alta, stando ad indicare che il 97% delle immagini classificate come anomale lo è effettivamente. Il restante 3% è dato da immagini di *background* classificate "volutamente" in modo errato: queste sono infatti conseguenza dell'introduzione della *pfa* nella fase di test che permette alla rete di sbagliare alcune immagini di traffico normale, riducendo quindi la "*Precision*". Tuttavia questa tecnica porta ad un notevole incremento della metrica "*Recall*" e quindi del numero di immagini anomale riconosciute dalla rete e classificate come tali.

		All Attacks			
		F1Score	Accuracy	Precision	Recall
PFA 0.05%	dots pfa 0.05%	0.01787	0.8518	0.5556	0.00908
	connected dots pfa 0.05%	0.8128	0.9526	0.9828	0.693
	connected dots wBCE pfa 0.05%	0.8256	0.9551	0.9761	0.7154
PFA 0.01%	dots pfa 0.01%	0.01787	0.8518	0.5556	0.00908
	connected dots pfa 0.01%	0.5985	0.9143	0.9823	0.4303
	connected dots wBCE pfa 0.01%	0.7968	0.9494	0.9866	0.6682

Tabella 4.10: Risultati di test su immagini ideali + non ideali con tutti gli attacchi

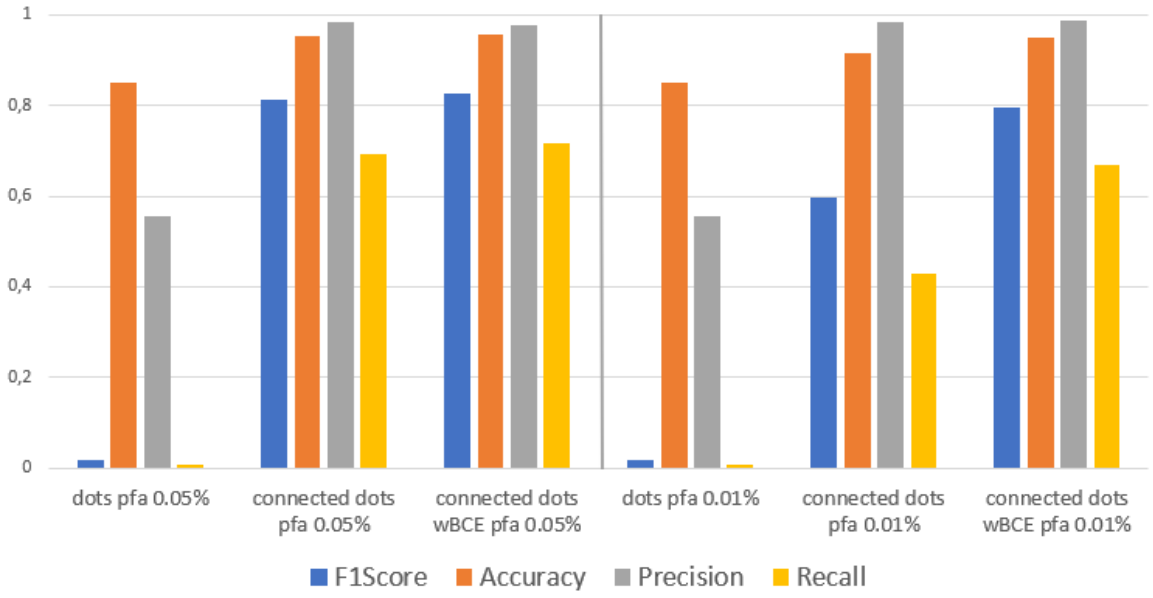


Figura 4.10: Grafico risultati di test su immagini ideali + non ideali con tutti gli attacchi

Capitolo 5

Conclusions

Questa tesi vuole proporre una nuova tecnica di rappresentazione 2D del traffico di rete, sviluppando un sistema in grado di rilevare l'eventuale presenza di anomalie in modo affidabile. Una volta terminato l'addestramento sulla rappresentazione precedente definita "sparsa", le prestazioni della rete sono risultate essere molto scarse: nella maggior parte dei casi venivano riconosciute meno dell'1% delle immagini anomale.

La rappresentazione connessa proposta in questa tesi ha portato invece a diversi benefici. I test sperimentali hanno dimostrato come questo nuovo metodo porti ad un notevole aumento delle prestazioni della rete, che riesce ora a classificare correttamente circa il 70% di immagini relative a traffico di rete con una generica anomalia. Successivamente, l'applicazione di BCE pesata e *batch normalization* all'architettura CNN utilizzata (*Autoencoder*), ha permesso di ottenere un miglioramento generale delle prestazioni, questa volta più bilanciato e distribuito tra tutte e 4 le metriche prese in considerazione.

Nel caso specifico, quindi con attacco Scan11, Scan44 e DoS11, il modello risulta essere affidabile classificanndo correttamente la maggior parte delle immagini. Il caso peggiore rimane tuttavia quello con attacco DoS53 dove la rete fatica a distinguere immagini anomale da quelle di *background*. Questo è dovuto sia al dataset da cui sono stati presi i dati relativi al traffico di rete, sia alla tecnica tramite cui questi vengono convertiti in immagini 2D. Sicuramente le prestazioni del sistema di riconoscimento delle anomalie sviluppato in questa tesi hanno ancora margine di miglioramento. La finestra temporale durante la quale si osserva e si raccolgono i dati relativi al traffico di rete può essere ad esempio aumentata a due o più secondi, garantendo così una maggior quantità di informazione. La tecnica che permette di tradurre i dati di rete in rappresentazione 2D inoltre può essere modificata al fine di trovarne una nuova versione ottimale che permetta di aumentare la capacità delle immagini di descrivere al meglio

il traffico di rete. I risultati ottenuti in questa tesi risultano essere comunque molto buoni e dimostrano come, con un'architettura CNN più semplice e al contempo una rappresentazione 2D migliorata, si riescano ad ottenere importanti incrementi delle prestazioni.

Bibliografia

[1] Dor Bank, Noam Koenigstein, and Raja Giryes. Autoencoders, 2021. arXiv:2003.05991.

[2] José Camacho, Alejandro Pérez-Villegas, Pedro García-Teodoro, and Gabriel Maciá-Fernández. Pca-based multivariate statistical network monitoring for anomaly detection. 01 2016.

[3] Sofia Casarin, Sara Baldoni, Marco Carli, Pietro Zanuttigh, and Federica Battisti. Unsupervised network anomaly detection by learning on 2d data representations. In *2022 9th Swiss Conference on Data Science (SDS)*, pages 53–58, 2022. doi:10.1109/SDS54800.2022.00016.

[4] Mehmet Celenk, Thomas Conley, John Willis, and James Graham. Predictive network anomaly detection and visualization. *IEEE Transactions on Information Forensics and Security*, 5(2):288–299, 2010. doi:10.1109/TIFS.2010.2041808.

[5] I. L. Dryden and K. V. Mardia. Multivariate shape analysis. *Sankhyā: The Indian Journal of Statistics, Series A (1961-2002)*, 55(3):460–480, 1993. URL: http://www.jstor.org/stable/25050954.

[6] Anukool Lakhina, Mark Crovella, and Christophe Diot. Diagnosing network-wide traffic anomalies. *SIGCOMM Comput. Commun. Rev.*, 34(4):219–230, aug 2004. doi:10.1145/1030194.1015492.

[7] Aleksandar Lazarevic, Levent Ertöz, Vipin Kumar, Aysel Ozgur, and Jaideep Srivastava. A comparative study of anomaly detection schemes in network intrusion detection. volume 3, 05 2003. doi:10.1137/1.9781611972733.3.

[8] Gabriel Maciá-Fernández, José Camacho, Roberto Magán-Carrión, Pedro García-Teodoro, and Roberto Therón. Ugr'16: A new dataset for the evaluation of cyclostationarity-based

network idss. *Computers & Security*, 73:411–424, 2018. URL: https://www.sciencedirect.com/science/article/pii/S0167404817302353, doi:https://doi.org/10.1016/j.cose.2017.11.004.

[9] Andrzej Maćkiewicz and Waldemar Ratajczak. Principal components analysis (pca). *Computers & Geosciences*, 19(3):303–342, 1993. URL: https://www.sciencedirect.com/science/article/pii/009830049390090R, doi:https://doi.org/10.1016/0098-3004(93)90090-R.

[10] Keiron O'Shea and Ryan Nash. An introduction to convolutional neural networks, 2015. arXiv:1511.08458.

[11] Cheng Wang and Hangyu Zhu. Wrongdoing monitor: A graph-based behavioral anomaly detection in cyber security. *IEEE Transactions on Information Forensics and Security*, 17:2703–2718, 2022. doi:10.1109/TIFS.2022.3191493.

[12] Yang Xiang, Ke Li, and Wanlei Zhou. Low-rate ddos attacks detection and traceback by using new information metrics. *IEEE Transactions on Information Forensics and Security*, 6(2):426–437, 2011. doi:10.1109/TIFS.2011.2107320.

Ringraziamenti

A conclusione di questo elaborato, penso sia doveroso dedicare uno spazio alle persone che mi hanno supportato durante questi anni.

Ringrazio innanzitutto la mia relatrice, la Prof.ssa Battisti che mi ha permesso di svolgere questo studio e di acquisire conoscenze su un argomento che mi appassiona profondamente. Un grazie va anche a Michael Neri per la sua infinita pazienza, per i consigli indispensabili che mi ha dato durante tutto lo svolgimento della tesi e per l'enorme quantità di conoscenze che mi ha trasmesso.

Un grazie speciale va ai miei genitori e mia sorella, i quali mi hanno sopportato in tutti questi anni. Senza il loro immenso sostegno, non avrei mai potuto raggiungere questo importante risultato.

Ringrazio Serena, la mia migliore amica che mi è rimasta sempre accanto e con cui ho condiviso ogni gioia ed ogni difficoltà. Sei una delle persone più importanti della mia vita e senza di te portare a termine questo percorso sarebbe stato impossibile.

Un ringraziamento anche ad Anna, un'amica fantastica conosciuta durante il percorso di studi. Grazie per avermi sopportato in questi anni, per avermi sempre aiutato e per aver creduto in me più di quanto facessi io.

Un grazie anche a Fabio e Giulia, due amici stupendi con cui spero di continuare a condividere i ricordi più belli. Grazie per ascoltarmi e riuscire sempre a farmi sorridere.

Infine un grazie a tutti i miei amici per il supporto e per rendere ogni giorno migliore.